UM CONJUNTO BÁSICO DE MEDIDAS DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE PARA A IMPLEMENTAÇÃO DO CMM IV

LÚCIA HELENA MAZONI COUTO

Universidade Federal do Rio De Janeiro – UFRJ
Instituto de Matemática - IM
Núcleo de Computação Eletrônica - NCE

Dissertação de Mestrado

Grau: Mestrado em Informática

Orientador: Eber Schmitz
Ph.D. em Ciência da Computação

Rio de Janeiro - RJ
Março/2004

UM CONJUNTO BÁSICO DE MEDIDAS DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE PARA A IMPLEMENTAÇÃO DO CMM IV

LÚCIA HELENA MAZONI COUTO

Dissertação de Mestrado submetida ao corpo docente do Instituto de Matemática e Núcleo de Computação Eletrônica (IM/NCE) – Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção do grau de Mestre.

Aprovada por:

_____

Prof. Eber Schmitz, Ph. D., NCE/DCC UFRJ – Orientador (eber@nce.ufrj.br)

_____

Prof. Márcio de Oliveira Barros, DSc, UNI RIO (marcio.barros@uniriotec.br)

_____

Prof.ª Maria Luiza Machado Campos, Ph.D., NCE/DCC UFRJ (mluiza@nce.ufrj.br)

Rio de Janeiro - RJ

Março/2004

Couto, Lúcia Helena Mazoni Couto
Um Conjunto Básico de Medidas do Processo de Desenvolvimento de Software para a implementação do CMM IV.
Rio de Janeiro: UFRJ/IM – NCE, 2004

viii, 78p.
Dissertação (Mestrado) – Universidade Federal do Rio de Janeiro, IM/NCE, 2004.
1. Medidas. 2. Métricas. 3 – Controle Estatístico de Processos. 4. CMM. 5. SPC.

AGRADECIMENTOS

A Deus, por cada minuto de graça concedido para a concretização deste trabalho.

Ao meu marido Alexandre, por sua paciência comigo nos piores momentos deste caminho, e que muitas vezes ficou com a responsabilidade de cuidar de nosso filho durante este caminho.

Ao meu filho Felipe, por existir e alegrar minha vida mesmo nos momentos difíceis.

Aos meus pais, que me apoiaram durante toda a vida para que meus sonhos fossem realizados, sempre com palavras e atos de carinho e incentivo.

À minha amiga Ana, por sua franqueza ao mostrar que estava se tomando um rumo errado na tese, e por sua disponibilidade de ajudar.

À Fátima, Geane e Lygia, pelo incentivo e pela confiança.

À Electronic Data Systems do Brasil, pela confiança concedida para que eu pudesse aprimorar meus conhecimentos.

Ao professor Eber Schmitz, meu orientador, pela sabedoria em saber extrair o melhor de cada aluno e de cada assunto e pelo grande incentivo para a concretização deste trabalho.

Ao Paulo Pereira, que foi fundamental na implementação do repositório de dados do conjunto proposto.

A todos os demais que participaram direta ou indiretamente neste trabalho.

RESUMO

COUTO, Lúcia Helena Mazoni. **Um Conjunto Básico de Medidas do Processo de Desenvolvimento de Software para a implementação do CMM IV.**

Orientador: Prof. Eber Schmitz. Rio de Janeiro: UFRJ/IM/NCE, 2004. Diss.

O *Capability Maturity Model* (CMM) descreve um caminho evolucionário para uma organização de software, desde um processo imaturo - ad-hoc (nível I) - até um processo maduro, de alta qualidade (nível V). A partir do nível IV é exigido que se faça gerenciamento quantitativo dos processos. Todavia, muitas vezes as medidas coletadas no nível III não são armazenadas no nível de detalhe adequado, nem são armazenadas de maneira apropriada para que seja possível utilizá-las para controlar estatisticamente os processos.

Neste trabalho são apresentados os conceitos envolvidos no CMM, especialmente os relacionados ao nível IV do CMM. Também é apresentada uma introdução aos conceitos do Controle Estatístico de Processos (*Statistical Process Control - SPC*), que devem ser utilizados para se atingir os níveis IV e V do CMM.

Partindo da premissa de que o próprio CMM delineia as principais medidas relacionadas ao processo de desenvolvimento de software, este trabalho fez um levantamento nas áreas de processos chaves (*KPA*s) do CMM relacionadas à engenharia de software, buscando definir um conjunto básico de medidas do processo de desenvolvimento de software que pudesse servir de base para a implementação do controle quantitativo de processos, passo *sinequanon* para se atingir os níveis IV e V do CMM.

Através de um estudo de caso, no qual este conjunto de medidas foi implementado em um projeto que participou da certificação de uma organização como nível IV do CMM, são levantados indícios de que este conjunto pode servir de base para a implementação do gerenciamento quantitativo e a melhoria contínua do processo de desenvolvimento de software de uma organização, pressuposto para se atingir os níveis IV e V do CMM.

ABSTRACT

COUTO, Lúcia Helena Mazoni. **Basic Software Engineering Measures to implement CMM IV**

Orientador: Prof. Eber Schmitz. Rio de Janeiro: UFRJ/IM/NCE, 2004. Diss.

The Capability Maturity Model (CMM) describes an evolutionary path to a software organization. At level IV the model requires quantitative management. However, even the organizations that achieved level III sometimes do not collect measures at the adequate level, nor store the measures in an adequate way.

In this work are presented fundamental concepts related to CMM and CMM IV, as well as an introduction to Statistical Process Control (SPC) concepts.

Using as assumption that the CMM might describe the most important measures related to software engineering, this work analyzed the Key Process Areas related to software engineering in order to define a basic set of measures that could be used to implement statistical process control in software organizations that are trying to achieve CMM levels IV and V.

By implementing this set of measures in a project that was audited when its organization was certified at CMM level IV, it was possible to find evidences that this set could be used by organizations as a basis to implement quantitative management process and process improvement, which are fundamental activities required to achieve CMM levels IV and V.

Lista de Siglas

BSCA	*Blue Shield of Califórnia* (neste contexto, a parte do projeto de manutenções desenvolvida pelo RioSC)
CL	Centerline (Linha central de controle do processo ou média)
CMM	*Capability Maturity Model – Modelo de Maturidade de Capacitação em Software*
CTQ	Critical to Quality
DoD	Departamento de Defesa (EUA)
IM	Instituto de Matemática
KPA	Key Process Area (Áreas de processo chaves)
LCL	*Lower Control Limit (*Limite Inferior de Controle do processo*)*
LSL	*Lower Specification Limit* (limite inferior do processo requerido pelo cliente*)*
NCE	Núcleo de Computação Eletrônica
RioSC	*Rio Solution Centre* – Fábrica de software da EDS Brasil no Rio de Janeiro
SEI	*Software Engineering Institute*, da Universidade Carnegie Mellon
SPC	*Statistical Process Control* (controle estatístico do processo)
UCL	*Upper Control Limit* (Limite Superior de Controle do processo)
UFRJ	Universidade Federal do Rio de Janeiro
USL	*Upper Specification Limit* (limite superior do processo requerido pelo cliente)

Lista de Ilustrações

Lista de Tabelas

Sumário

1 INTRODUÇÃO

1.1 POSICIONAMENTO DO TRABALHO

O *Capability Maturity Model (CMM)* descreve um caminho evolucionário para uma organização de software, desde um processo imaturo - ad-hoc (nível I) - até um processo maduro, de alta qualidade (nível V) (Paulk et al, 1993, p. O-13).

O CMM, inicialmente criado para avaliar as empresas prestadoras de serviço para o DOC (Departamento de Defesa dos Estados Unidos) (Paulk et al., 1993, p.2), tem sido cada vez mais exigido na indústria da tecnologia da informação como demonstração da maturidade das empresas prestadoras de serviços nesta área. No Brasil, já existem algumas empresas certificadas nos níveis II e III do CMM. A implementação do nível IV, porém, ainda é um terreno praticamente inexplorado no Brasil (Ministéria da Ciência e Tecnologia).

Em seu artigo "*Lessons Learned in Attempting to Achieve Software CMM Level 4*" (2001, p.30), Al Florence descreve uma tentativa mal-sucedida de se certificar no nível IV do CMM, e afirma que a implementação do CMM IV é uma mudança drástica em relação aos paradigmas usados no nível III. Florence menciona as principais diferenças:

- Os níveis II e III do CMM descrevem coisas a *serem feitas* para se desenvolver um bom software, enquanto que o nível IV vai muito adiante disso, sendo para organizações que realmente desejam percorrer o caminho para a melhoria contínua dos processos.

- O nível III exige apenas a aplicação de conhecimentos já existentes de engenharia de software e gerenciamento de projetos., ao passo que no nível IV todo um conhecimento quantitativo e de estatístico necessita ser adquirido.

- O maior foco do nível III é a organização e no nível IV o foco se volta outra vez ao projeto.

- O nível III requer que os processos sejam institucionalizados, ao passo que o nível IV requer que a capacidade dos processos seja conhecida e controlada quantitativamente.

- No nível III é requerido que a garantia de qualidade seja institucionalizada. O nível IV requer que sejam estabelecidos planos para o atendimento de objetivos de qualidade, e que o progresso destes planos sejam gerenciados quantitativamente.
- Nos níveis II e III do CMM, uma empresa tem que coletar e usar medidas basicamente para determinar status e corrigir problemas, como por exemplo o status das atividades de gerenciamento de recursos, das atividades de controle de configuração, de planejamento de software, etc... No nível IV é requerido que as medidas sejam analisadas quantitativamente para controlar o desempenho dos processos e que ações imediatas sejam tomadas para endereçar os problemas.

Na época em que foi definido o CMM, os autores reconheceram que as características definidas para os níveis IV e V foram elaboradas por similaridades com outras engenharias, já que na época poucas organizações de software usavam o controle estatístico de processos para software: o SPC era mais usado em controle de qualidade de fábricas (Paulk, Curtis et al, 1993, p. 20).).

Como visto, a distância entre os níveis III e IV do CMM não é pequena, e um dos pontos que podem diminuir ou aumentar esta distância são as medidas armazenadas no nível III. Até o nível III do CMM, como o objetivo da coleta de medidas é determinar status e corrigir problemas de diferentes áreas do projeto, é comum que estas medidas sejam levantadas e mantidas em diferentes lugares e em diferentes níveis de detalhe. Por causa disso, ao começar a caminhar para o nível IV, dependendo da estruturação e do nível de detalhe das medidas colhidas no nível III, pode-se perceber a necessidade de recomeçar o processo de coleta de medidas no nível de detalhe apropriado, o que pode ser muito custoso e pode tornar mais longo o caminho para se chegar ao nível IV do CMM.

1.2 OBJETIVOS

Para fazer a análise quantitativa dos processos o CMM IV prevê que seja utilizada uma base de dados de medidas sobre o processo de desenvolvimento de software, comum à organização, para a coleta e análise dos dados disponíveis dos projetos de software (Paulk, Weber et al., *Ibid*, p. O-16).

Visando facilitar o processo de definição de uma base de dados de medidas sobre o processo de desenvolvimento de software, que possa ser utilizada desde o nível III do

CMM de forma a facilitar a implementação do nível IV, este trabalho propõe um conjunto básico de medidas do processo de desenvolvimento de software, que esteja num nível de detalhe suficiente para auxiliar o processo de implementação do controle quantitativo dos processos em uma organização.

Em cada área de processo chave (KPA), o CMM menciona medidas que são inerentes a ela. Também menciona exemplos de medidas como informação suplementar, assumindo que a variabilidade dos projetos pode levar à necessidade de diferentes medidas e estratégias (Paulk et al, 1993, p. O-47). Todavia, é possível levantar a hipótese de que é possível definir um conjunto básico de medidas do processo de desenvolvimento que enderece os dados mínimos necessários para que um projeto possa controlar quantitativamente seus processos. Esta hipótese é assumida neste trabalho porque ao mencionar a evolução do processo de coleta e análise de medidas (p. O-63), é explicado que no nível IV a organização deve definir um conjunto de medidas baseadas no processo de desenvolvimento de software padrão da organização, e que o projeto depois pode acrescentar a este conjunto medidas que sejam específicas para o projeto. Se uma organização, que pode ter diferentes tipos de projetos, pode definir um conjunto de medidas padrão para seus projetos, então o pressuposto de que um conjunto de medidas do processo de software possa ser usado como padrão inicial por organizações que estão no processo de amadurecimento é assumido como válido.

Neste trabalho é assumido também como premissa que se podem encontrar no próprio modelo CMM os elementos essenciais que devem fazer parte de um conjunto mínimo de medidas do processo de desenvolvimento de software, através da análise das *KPAs* do *CMM* relacionadas à engenharia de software (Paulk at al, *ibid*, p. O-35).

O objetivo do conjunto é ser abrangente, envolvendo os diferentes aspectos do desenvolvimento de software, e que através da implementação deste conjunto as organizações tenham uma base estruturada para iniciar o processo de conhecimento do comportamento de seus processos, possam usar este conhecimento para implementar a melhoria contínua de seus processos, pressuposto para se atingir os níveis IV e V do CMM. Outro benefício que se pode levantar é que as organizações podem começar a coletar medidas de uma forma coerente, já desde o nível III do CMM.

Para avaliar se a utilização deste conjunto básico de medidas do processo de desenvolvimento de software é válida e realmente facilitaria a implementação do nível IV

do CMM, este foi implementado em um projeto de uma organização que estava em processo de implementação do CMM IV. A implementação deste conjunto de medidas em um projeto que estará iniciando o gerenciamento quantitativo de processos visa não apenas validar o conjunto, mas também mostrar as vantagens de se iniciar o processo de gerenciamento quantitativo com dados históricos abrangentes, armazenados em um nível de detalhe suficiente.

1.3 DELIMITAÇÃO DO ESCOPO

Muitas vezes a palavra medida é usada no mesmo contexto que a palavra métrica. Como este trabalho propõe-se a estabelecer um conjunto de medidas, torna-se necessário delimitar o conceito de medida que será utilizado, diferenciando-o do conceito de métrica. *Medida* provê uma indicação quantitativa da extensão, quantidade, dimensão, capacidade ou tamanho de certos atributos de um produto ou processo. Quando um dado individual é coletado, uma medida foi estabelecida. Uma *métrica* de software relaciona a medida individual com outras de alguma forma, como por exemplo o número médio de erros encontrados por pessoa/hora gasta em revisões (Pressman, 2000). De forma simplificada, *medidas* seriam atributos simples de um produto ou processo, *métricas* seriam atributos calculados, ou relações entre medidas.

Considerando as perspectivas mencionadas, o conjunto proposto visa reunir **medidas** sobre o **processo** de desenvolvimento de software para que, correlacionando-se estas medidas, seja possível analisar as **métricas** do processo, gerando um maior conhecimento do processo de desenvolvimento de software.

O conjunto proposto de medidas do processo de desenvolvimento de software pode ser customizado para atender às peculiaridades de cada organização – as fases são customizáveis, bem como a nomenclatura de tipos de defeitos, etc... Os valores propostos devem ser considerados apenas como sugestões.

Analisando as KPAs do modelo CMM, pode-se verificar que várias medidas são solicitadas. Não está no escopo deste trabalho levantar medidas como as organizacionais ou de treinamento. O propósito desde trabalho é mapear o processo de desenvolvimento de software, que é o propósito do nível IV do CMM: "... estabelecer uma compreensão quantitativa tanto sobre o processo de desenvolvimento de software quanto sobre os softwares produzidos" (Paulk *et al*, 1993, p.O-23).

Este conjunto de medidas não tem a intenção de ser a única fonte a ser utilizada no gerenciamento de projetos, pois faltam informações como riscos, problemas, etc... Contudo, pode ser usado como base para a análise estatística dos processos de engenharia de software, de identificação de áreas a sofrerem melhorias de processos, em linha com a aderência aos requisitos dos níveis IV e V do CMM. Logicamente como fruto destas análises estatísticas podem ser identificados riscos de forma mais cedo, mas este conjunto não serve de repositório para eles.

Na implementação do CMM IV, não existe a intenção de que se coloquem todas estas medidas sob controle estatístico. Estas medidas serão opções para verificar que sub-processo mais influencia o processo que se quer melhorar, de acordo com os objetivos da organização.

1.4 ORGANIZAÇÃO DO TRABALHO

Esta dissertação está organizada em 5 capítulos, objetivando apresentar aspectos teóricos e práticos relacionados ao desenvolvimento da mesma:

- Capítulo 1, Introdução – Apresenta e situa esta dissertação em um contexto global, por intermédio da exposição de sua motivação, objetivo e escopo.
- Capítulo 2, *Capability Maturity Model* – Neste capítulo é situado o contexto do trabalho, apresentando-se como se iniciou o CMM, como ele se estrutura e requisitos específicos para o nível IV do CMM.
- Capítulo 3 – Introdução ao Controle Estatístico de Processos (SPC) - Faz uma introdução ao SPC, definindo seus principais conceitos e ferramentas, além de exemplo de aplicação do SPC em uma organização de software.
- Capítulo 4, Definição do Conjunto de Medidas do processo de desenvolvimento de software – Analisando as KPAs do CMM relacionadas à engenharia de software, define o conjunto básico de medidas do processo de desenvolvimento de software.
- Capítulo 5, Estudo de Caso – Apresenta a implementação do conjunto proposto e sua aplicação no gerenciamento quantitativo do processo.
- Capítulo 6, Conclusões – Relata, de forma sucinta, as principais contribuições desta dissertação e apresenta algumas sugestões para trabalhos futuros.

2 CAPABILITY MATURITY MODEL

2.1 ORIGEM DO CMM

O *Software Engineering Institute* (SEI) foi fundado pelo governo dos Estados Unidos junto com a Universidade *Carnegie Mellon* com o objetivo de melhorar o nível das práticas na engenharia de software (Perdue, 2000). Com a assistência da *Mitre Corporation*, a SEI começou a desenvolver a estrutura de maturidade do processo que ajudaria as organizações a melhorar seus processos de software. Esse esforço foi iniciado em resposta a uma solicitação do Departamento de Defesa do governo norte-americano (DoD) de se desenvolver um método para avaliar a capacitação dos fornecedores de software contratadas pela Força Aérea Norte Americana (Paulk et al., 1993, p.2).

O objetivo do *Capability Maturity Model* (CMM), proposto pela SEI, era guiar as organizações de software no processo de seleção das estratégias de melhoria, determinando a maturidade atual do processo e identificando as questões mais críticas para a qualidade e melhoria do processo de software (Paulk et al., Ibid). Embora, historicamente, o CMM tenha surgido no contexto de grandes empresas de desenvolvimento de software contratadas pelas Forças Armadas dos EUA para projetos militares, tem-se verificado que seus princípios são válidos para todo tipo de projetos de software.

O *CMM* avalia a maturidade das organizações de software em cinco níveis, conforme ilustrado na figura 2.1.

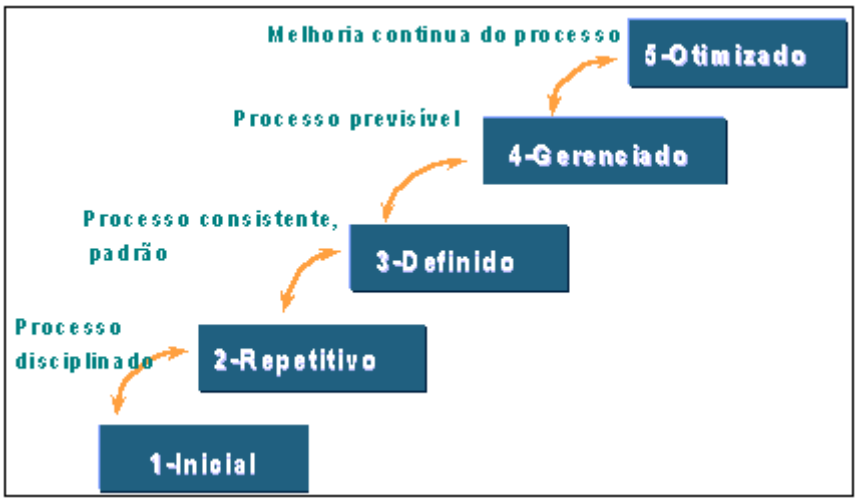

Figura 2-1- Níveis do CMM (baseado em Paulk et al, 1993, p. O-13)

No artigo "The *Capability Maturity Model for Software*" (Paulk, Curtis et al, 1993, p.7-10), Mark Paulk descreve como se caracterizam as organizações em cada nível de maturidade:

1) Inicial - O processo de software é caracterizado como "*ad hoc*" e até mesmo ocasionalmente caótico. Poucos processos são definidos e o sucesso depende de esforço individual.

2) Repetitivo - Os processos básicos de gerenciamento de projeto são estabelecidos para acompanhar custo, cronograma e funcionalidade. A necessária disciplina do processo existe para repetir sucessos anteriores em projetos com aplicações similares.

3) Definido - O processo utilizado para as atividades de gerenciamento de projetos e engenharia de software é documentado, padronizado e integrado em um processo de software padrão para a organização. Todos os projetos utilizam uma versão aprovada do processo de software padrão para desenvolver e manter software.

4) Gerenciado - Medidas detalhadas do processo de software e da qualidade do produto são realizadas. O processo e os produtos de software são quantitativamente compreendidos e controlados.

5) Otimizado- A melhoria contínua do processo é propiciada pelo *feedback* quantitativo do processo e pela introdução de novas soluções tecnológicas.

2.2 ESTRUTURA DO CMM

O CMM é composto de 5 níveis de maturidade, descritos anteriormente. Com exceção do nível 1, todos os outros são compostos de várias áreas de processos chaves (*Key process areas* -KPAs). Cada KPA é organizada em 5 seções chamadas de *common features*. Cada *common feature* especifica as práticas chaves que buscam atingir o objetivo (goal) da KPA (Paulk et al, *ibid*, p. O-8) . Esta estrutura é representada pela figura 2.2.

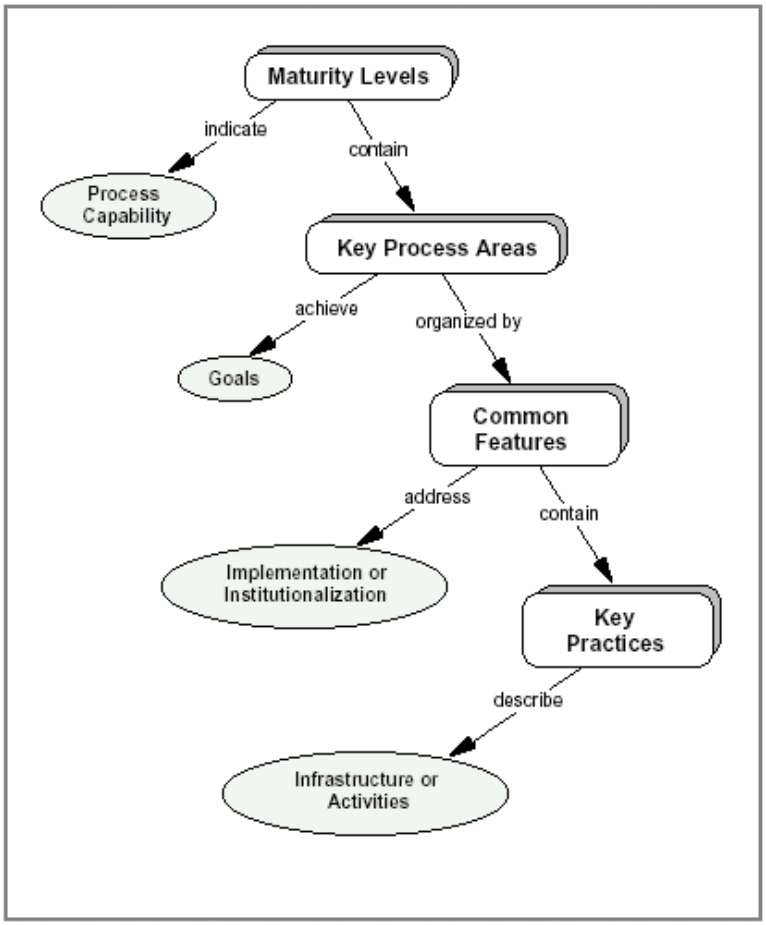

Figura 2-2- Estrutura do CMM (Paulk et al, ibid, p.O-9)

Os componentes desta estrutura são:

- **Nível de Maturidade** – Nível de evolução obtido ao se buscar um processo de software maduro.
- **Capacidade do processo** – Define quais são os possíveis resultados que podem ser atingidos ao se seguir o processo de software.
- **Áreas de processos chave (KPAs)** – conjunto de atividades relacionadas que coletivamente atingem um conjunto de objetivos importantes para se estabelecer a capacidade daquele nível de maturidade.
- **Objetivos** – Definem o escopo, fronteira e intenções de cada KPA. É usado para determinar se uma organização implementou realmente a KPA.
- **Common Features** – São atributos que indicam se a implementação e institucionalização de uma KPA é consistente. Em todas as KPAs são encontradas 5 *common features*, que agrupam as práticas de uma KPA: Comprometimento para

realizar, Habilidade para realizar, Atividades realizadas, Medição e Análise, Verificação da Implementação. A única *common feature* que endereça atividades a serem implementadas é a de "Atividades realizadas". Todas as demais endereçam fatores institucionais, que tornam o processo parte da organização, como treinamento, por exemplo.

A figura 2.3 mostra as áreas de processo chaves, também chamadas de *KPAs (Key process areas*) para cada nível de maturidade. Cada uma delas procura identificar itens que devem ser endereçados para se atingir o grau de maturidade proposto.

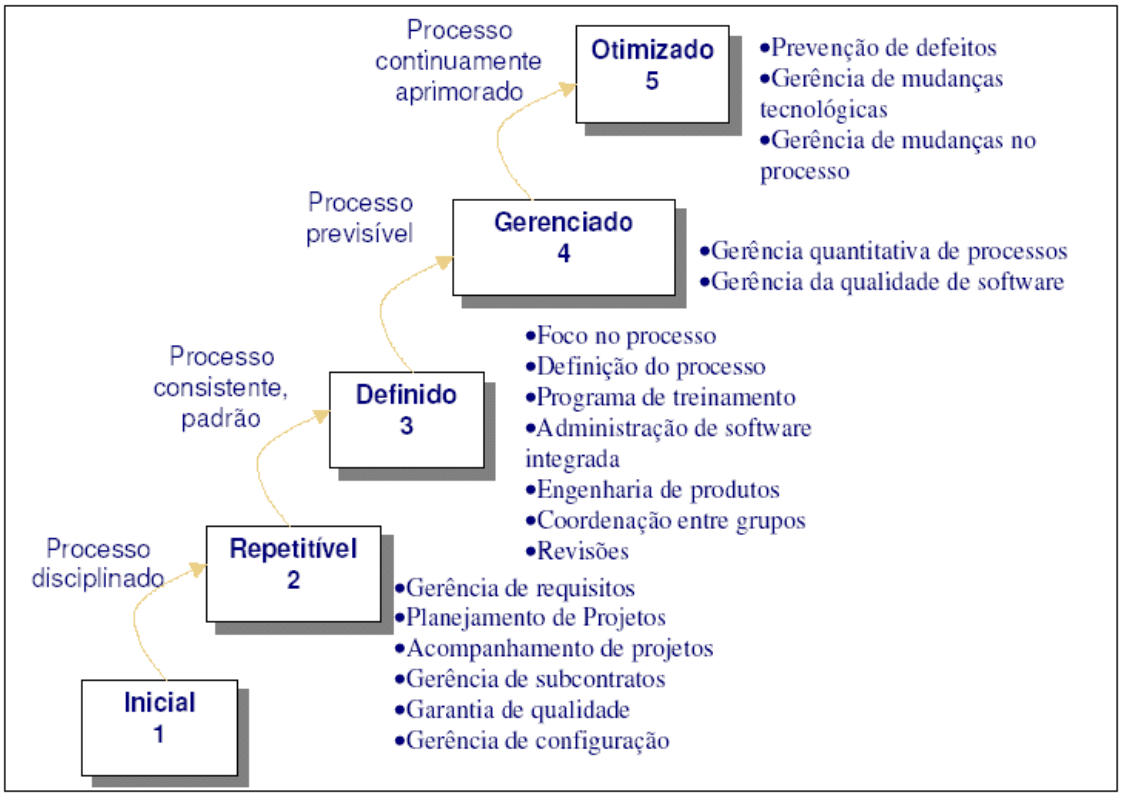

Figura 2-3 - KPAs por Nível de Maturidade (baseado em Paulk et al. Ibid, p. O-19)

2.3 NÍVEL IV DO CMM

O objetivo principal do nível IV do CMM é estabelecer um conhecimento quantitativo tanto do processo de software quanto dos produtos gerados por este processo. As duas KPAs do nível IV do CMM, "Gerenciamento Quantitativo de Processos" e "Gerenciamento da Qualidade de Software", são descritos no *artigo "Key Practices of the Capability Maturity Model, Version 1.1*", de Mark Paulk e outros (Paulk et al, 1993, p. L4-32).

O propósito da KPA de **Gerenciamento Quantitativo de Processos** é que o desempenho do processo de software do projeto seja controlado de forma quantitativa.

O Gerenciamento Quantitativo de Processos estabelece que é necessário (1) estabelecer metas para o desempenho do processo de desenvolvimento de software, (2) levantar medidas do desempenho do processo, (3) analisar estas medidas e (4) fazer os ajustes necessários para manter o desempenho do processo dentro de limites aceitáveis. Uma vez que o processo de desenvolvimento de software esteja estabilizado dentro de limites aceitáveis, o processo e os limites são estabelecidos como *baseline* e usados para controlar quantitativamente o desempenho do processo (Paulk, Weber et al., *ibid*, pg L.4-1).

Ainda nesta KPA, é visto que a organização deve coletar dados sobre o desempenho dos projetos para determinar a capacidade do processo usado, ou seja, quais são os valores esperados ao se seguir determinado processo. Uma vez sendo a capacidade do processo conhecida, o projeto a usa para rever os objetivos estabelecidos e analisar o desempenho do processo (Paulk, Weber et al., *ibid*, pg L.4-2).

O **Gerenciamento da Qualidade do Software** envolve (1) a definição de objetivos de qualidade para os produtos do projeto de software, (2) o estabelecimento de planos para se atingir estes objetivos e (3) o monitoramento e ajuste dos planos, produtos, atividades e objetivos para satisfazer as necessidades e desejos do cliente (Paulk et al., *ibid*, pg L.4-19).

O propósito da KPA de Gerenciamento da Qualidade de Software é que a organização desenvolva um conhecimento quantitativo sobre a qualidade dos produtos de software desenvolvidos e que atinja os objetivos de qualidade da organização. Estes objetivos devem ser baseados nas necessidades e prioridades da organização e dos clientes (Paulk, Weber et al., *ibid*, pg L.4-23).

Sobre a quantidade de objetivos a serem controlados quantitativamente, o CMM define que o plano de qualidade de software deve conter os objetivos de qualidade "*high-leverage*", definindo estes objetivos como sendo aqueles que trarão a maior satisfação do cliente ao menor custo (Paulk, Weber et al., *ibid*, pg L.4-26). Desta forma fica evidenciado que o projeto/organização poderá definir vários objetivos de qualidade, mas que serão priorizados de acordo com o benefício/custo de implementá-los.

Também nesta KPA é definido que deve ser verificado se a capacidade do processo permite que se atinjam os objetivos priorizados (Paulk et al., *ibid*, pg L.4-23).

2.4 CONCLUSÃO DO CAPÍTULO

Para melhor conhecimento do *Capability Maturity Model*, este capítulo tratou dos seus principais aspectos, por meio da exposição de sua origem, objetivos, níveis em que ele classifica uma organização, estrutura e áreas chaves de processos (KPAs).

Foram também detalhados neste capítulo os objetivos da KPAs do nível IV do CMM, a saber o Gerenciamento Quantitativo de Processos e o Gerenciamento da Qualidade do Software, que pressupõem a implementação do controle estatístico de processos.

Neste enfoque, o próximo capítulo faz uma introdução ao SPC (*Statistical Process Control*), ou Controle Estatístico de processos, para que se possa entender seus principais conceitos e auxiliar na interpretação do estudo de casos.

3 CONTROLE ESTATÍSTICO DE PROCESSOS (SPC)

Na década de 20 Walter S. Shewhart definiu o *Statistical Process Control* – SPC (Controle Estatístico do Processo) como uma forma metódica de monitorar e medir se os itens fabricados estavam de acordo com as especificações. Ele analisou que no sistema fabril não existia 100% de conformidade com as especificações, mas que em todo fenômeno existia um certo nível de variabilidade natural devido ao envelhecimento das máquinas, diferenças entre os componentes, diferentes capacidades dos trabalhadores, etc. Devido a isto, Shewhart concluiu que o desafio da qualidade era controlar a variabilidade entre os itens fabricados de forma que a variação ocorresse dentro de limites aceitáveis (Shewhart, 1931).

De uma forma simplificada, o SPC pode ser definido como a análise de um processo baseada em estatística e medidas sobre o desempenho do processo. O objetivo desta análise é identificar causas comuns e especiais de variação do desempenho do processo, de forma a manter o desempenho do processo dentro de limites (Gupta, 2002, p.4).

As técnicas definidas por Shewhart foram muito usadas na Segunda Guerra Mundial e mais tarde por Edwards Deming e outros para melhorar a qualidade de produtos, principalmente no Japão. Graças a esta utilização estas técnicas foram também adotadas em outras áreas de negócio, mais recentemente na área de software (Florac et al, 1997, p. 23).

A implementação dos níveis IV e V do CMM pressupõe a utilização de técnicas estatísticas para controlar e melhorar os processos. Usando os termos do Controle Estatístico do Processo (SPC), Paulk define que o foco do nível IV do CMM é remover as causas especiais de variação e o do nível V é sistematicamente tratar as causas comuns de variação do processo (Paulk, 1999, p.1). Neste capítulo serão abordados os principais conceitos e ferramentas mais utilizadas na implementação do SPC.

3.1 PROCESSO DE SOFTWARE

A definição de processo mais coerente com os princípios do Controle Estatístico de Processos (SPC) é a de Gabriel Pall (*apud* Florac et al, 1997, p.5): "um processo pode ser definido como a organização lógica de pessoas, materiais, energia, equipamentos e

procedimentos em atividades cujo objetivo é produzir um resultado definido". Esta definição de processo de software é demonstrada na figura a 3.1.

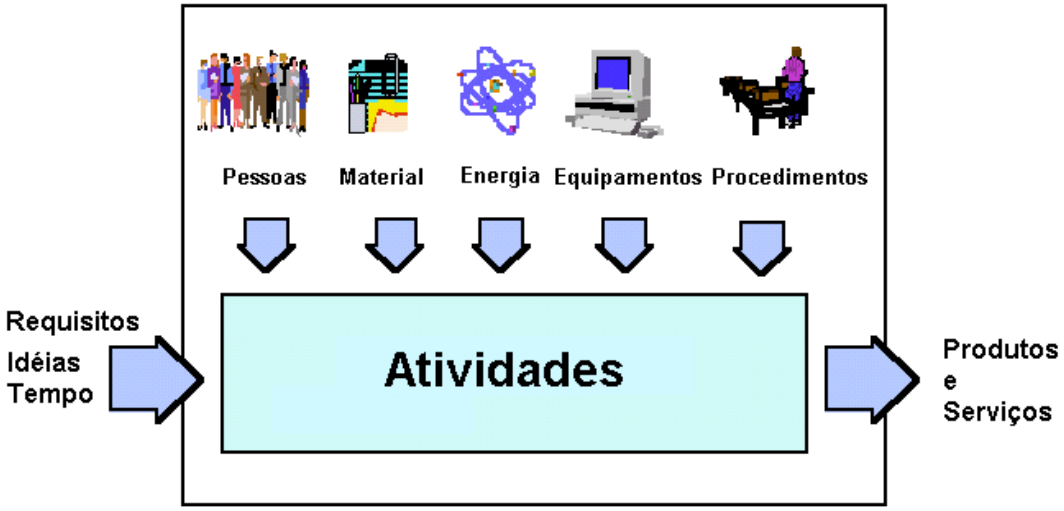

Figura 3-1 Definição de Processo (Florac, 1997, p. 6)

Este conceito de processo facilita a aplicação dos princípios estatísticos porque demonstra as diversas variáveis a serem levadas em consideração quando for necessário investigar causas de variação no processo (Florac et al, 1997, p.6).

3.2 CONTROLE DO PROCESSO

"Controlar um processo significa fazê-lo se comportar da forma que desejamos".

Willian Florac,1997

Segundo William Florac (1997, p.15), ter seus processos sob controle permite que as organizações prevejam resultados e produzam produtos que tenham as características desejadas pelo cliente. Florac considera que o controle é o ponto central para se atingir os objetivos de negócio que fazem uma organização de software bem sucedida.

Florac sugere os seguintes passos para se controlar um processo:

1. **Conhecer o desempenho do processo** – O primeiro passo para se controlar o processo é descobrindo como ele se comporta, que resultados produz, ou seja, conhecer o desempenho do processo. Neste passo inicial não se busca julgar o processo, apenas conhece-lo.
2. **Avaliar a estabilidade do processo** – Uma vez estando o desempenho do processo quantificado, é possível avaliar se o processo está se comportando de forma estável ou

não. Se o comportamento do processo varia imprevisivelmente no tempo, o processo não está sob controle. Para controlar um processo, primeiramente é preciso ter um processo cuja variabilidade seja estável. Se não há estabilidade não se podem prever os resultados que o processo pode produzir.

3. **Estabilizar o processo** – Se o desempenho do processo é errático e imprevisível, é necessário tomar ações para estabilizá-lo. Para definir que ações são necessárias, Florac sugere as seguintes perguntas:
   - O processo é suportado adequadamente, de forma que ele se tornará estável se for seguido fielmente?
   - O processo está sendo seguido fielmente, como definido?
   - A organização está preparada para seguir o processo?
4. **Avaliar a capacidade do processo** – Ter um processo estável não significa que o desempenho dele seja satisfatório. O processo também precisa ser capaz. Capacidade significa que as variações do processo no tempo estão dentro dos limites requeridos para se obter sucesso. As medidas da capacidade do processo relacionam o desempenho do processo às especificações que o processo deve satisfazer.
5. **Melhorar o processo** – Se o processo de software não for capaz de consistentemente satisfazer os requisitos do produto e as necessidades de negócio, ou se a organização precisa satisfazer demandas cada vez mais crescentes por qualidade, será necessário melhorar o desempenho do processo continuamente. Entender a capacidade de cada sub-processo que compõe o processo é o primeiro passo para progredir na melhoria do processo.

   A seguir será detalhado cada um destes conceitos.

3.2.1 DESEMPENHO DO PROCESSO

Segundo Florac (*ibid*, p.18), a chave para se medir o desempenho do processo é escolher medidas que sejam relevantes ao processo escolhido e objetivo a ser atingido. Para medir o desempenho do processo deve-se medir tantos atributos relativos ao produto e processo quanto possível, durante vários pontos no tempo, de forma a obter dados seqüenciais sobre seu comportamento.

Para que os dados de desempenho sejam bem definidos, eles devem satisfazer a três critérios:

1. **Comunicação** – O método usado para definir as medidas ou descrever os valores permite que outros saibam precisamente o que foi medido e o que se incluiu/excluiu dos resultados agregados? Quem usar os dados sabe como eles foram coletados, de forma que possam interpretar os resultados?
2. **Repetibilidade** – Se outra pessoa medir novamente o processo chegará nos mesmos números?
3. **Rastreabilidade** – As origens dos dados estão identificadas em termos de tempo, seqüência, atividade, produto, ambiente, ferramentas utilizadas para medir e quem coletou a informação. Informações que mostrem todo o contexto dos dados ajudam a identificar possíveis causas de variação.

3.2.2 ESTABILIDADE DO PROCESSO

Para entender o que é um processo estável, primeiro é importante entender a variabilidade natural de todo processo.

Qualquer processo de produção, independentemente de suas características, contém muitas fontes de variação. Por melhor ajustado que esteja, ele produzirá resultados que apresentarão diferenças entre si, podendo ser grandes ou pequenas. Esta variabilidade natural é o conjunto de efeitos cumulativos que são compostos de várias causas.

Walter Shewart foi o primeiro a identificar que a variabilidade dos processos tinha duas causas básicas: causas comuns e causas especiais (Shewhart, 1931, p.37).

- **Causas comuns de variação** – Causas comuns de variação são inerentes ao processo, podem ser consideradas como parte do ritmo normal do processo. Elas existem por causa da interação entre os vários elementos que compõem o processo (pessoas, máquinas, materiais, ambientes e métodos). É definida como uma fonte de variação que afeta a todos os valores individuais do processo. Elas são evidenciadas através de um padrão randômico mas que varia dentro de limites previsíveis.
- **Causas especiais de variação -** As causas especiais ou esporádicas são fatores geradores de variações que afetam o comportamento do processo de forma

> imprevisível, não sendo possível obter-se um padrão. A causa esporádica diferencia-se da comum pelo fato de produzir resultados totalmente discrepantes em relação aos demais valores. Ela é gerada por eventos que não fazem parte do processo normal (Ramos, 1997, p.31).

Quando todas as causas especiais de variação forem removidas, garantindo-se que elas não ocorram novamente, o processo é dito estatisticamente estável, ou sob controle. Neste caso a variação do processo é previsível, ou seja, somente causas comuns estão presentes (Florac, *ibid*, p.21).

Na figura 3.2 é dado um exemplo de gráfico linear, onde estão sendo acompanhados indicadores de defeitos. O gráfico mostra causas especiais e causas comuns, que quando eliminadas geram uma melhoria real no processo.

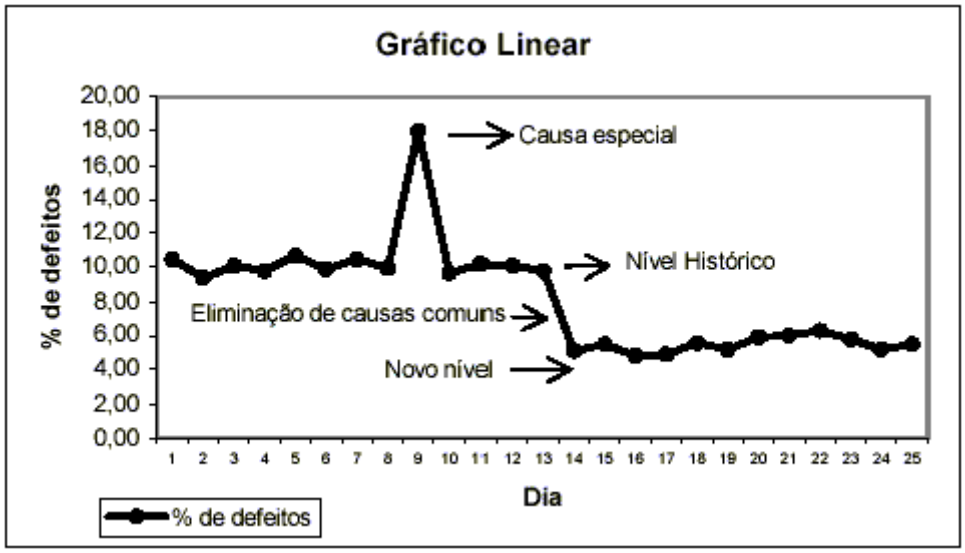

Figura 3-2- Causas Comuns e Especiais de Variação (Ramos, 1997, p.191)

A estabilidade do processo permite que se considere que os resultados do processo serão repetitíveis e portanto habilita a organização a prever o desempenho futuro e definir planos realistas, considerando este desempenho. A estabilidade pode ser considerada a base para se separar sinais de "barulho", de forma que se possa ter um controle efetivo sobre o processo e uma base sólida para a implantação de melhorias (Florac et al, 1997, p.5).

3.2.3 ESTABILIZAÇÃO DO PROCESSO

Para que um processo seja estável e previsível, le deve ser seguido consistentemente. Se um processo não é estável, a procura por causas especiais deve sempre começar pela verificação de se ele está sendo seguido ou não.

Florac define três aspectos que afetam significativamente o desempenho do processo (Florac et al, *ibid*, p.24-25):

1. **Habilidade para executar o processo** – Este aspecto permite que se entenda se a organização está apta e pronta para utilizar um processo. Se, por exemplo, os recursos do projeto não estão a par do processo, não foram treinados, ou não têm as ferramentas necessárias para executar um processo, provavelmente este processo não será executado adequadamente. Entender a habilidade da organização em executar os processos permite que se identifiquem ações adequadas a serem tomadas para remediar situações onde a falta de habilidade é a causa raiz da instabilidade do processo.
2. **Uso do processo definido –** A estabilidade do processo depende de que ele seja executado consistentemente como definido. Se as ferramentas requeridas não são utilizadas, ou se outros procedimentos são usados ao invés do procedimento definido no processo, é alta a probabilidade de que o comportamento do processo como um todo seja instável. Para se identificar se o processo está mesmo sendo seguido, deve-se fugir de questões fechadas do tipo "Você usa a ferramenta Y", ou "Você segue o procedimento X?". Para realmente entender como o processo é seguido, o ideal é usar questões abertas como "Quão freqüentemente você usa a ferramenta Y?", ou "Como você executa o procedimento X?", ou "Que treinamento você recebeu?".
3. **Supervisão e avaliação do processo –** Todos os processos são sujeitos a forças de entropia, isto é, deixados por si só, os processos tendem a piorar seu estado, indo do estado controlado para o caótico. Se não for feito um esforço consciente para sustentar os processos, reforçando seu uso correto e consertando os efeitos da entropia, existe a tendência de se perder o controle. A supervisão gerencial através de revisões periódicas do status do processo, auditorias formais nos processos e

comparações com outros projetos são as ferramentas utilizadas para que se sustente o processo.

3.2.4 CAPACIDADE DO PROCESSO

De forma simplificada, um processo é dito "capaz" se ele é um processo estável, cujo desempenho satisfaça os requisitos do cliente. Usando o exemplo da figura 3.3, pode-se comparar a estabilidade do processo com a precisão das flechas: elas estão distribuídas por todo o alvo ou mais em um lugar específico. Por outro lado, a capacidade do processo pode ser comparada com a acurácia das flechas, que avalia se elas estão atingindo o alvo proposto.

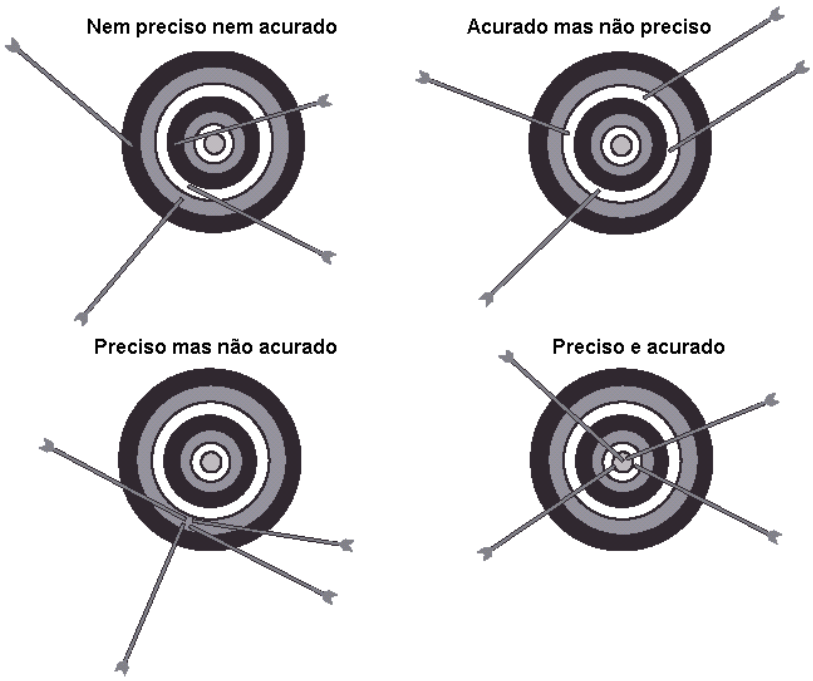

Figura 3-3- Precisão e Acurácia (adaptado de Montgomery, 1997, p.40)

De forma simplificada, examinar se um processo é capaz é responder a pergunta se ele está ou não atingindo suas metas.

Para exemplificar como a capacidade é avaliada, Florac utiliza o seguinte histograma da figura 3.4.

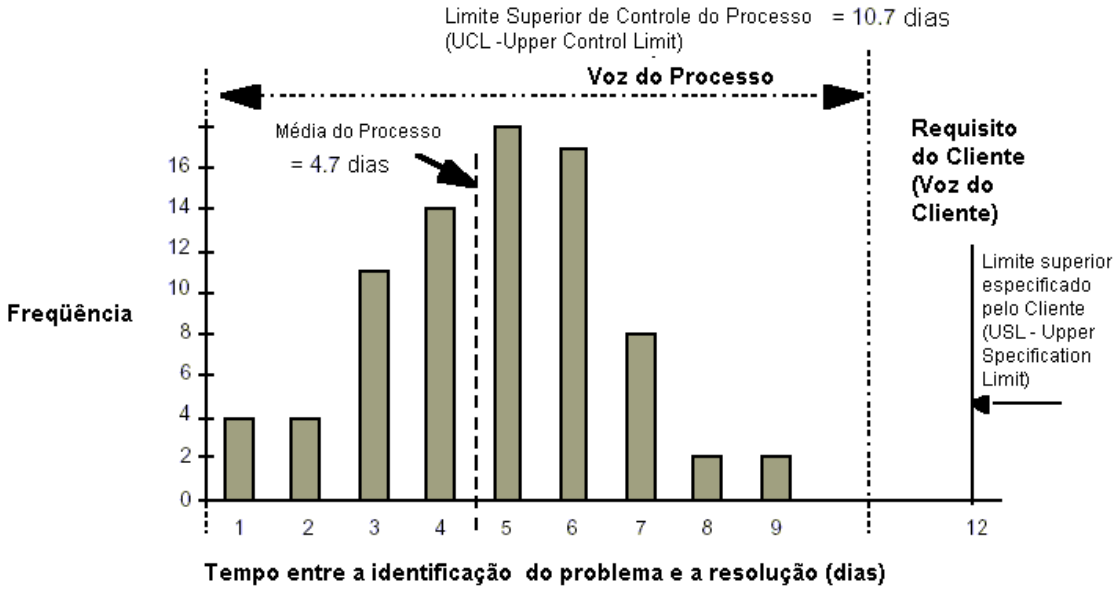

Figura 3-4 - Histograma refletindo a Capacidade do Processo (Florac et al, 1997, p. 28)

Este histograma mostra a freqüência de medidas obtidas de um processo estável. Como o processo é reconhecido como estável, a média e os limites de controle representam o que o processo está apto a fazer. No histograma a média e os limites de controle representam o que se chama de "a voz do processo".

Sobrepondo-se os requisitos do cliente ou de negócio no histograma, é possível comparar o que o processo "diz" do que o negócio diz que ele precisa. Estes requisitos (ou especificações) são também conhecidos como "a voz do cliente", são chamados de limites superiores e inferiores de especificação (USL – *Upper Specification Limit* e *LSL – Lower Specification Limit*) (Florac, *ibid*, p.29).

Se um ou dois dos limites de controle do processo estiverem fora dos limites de especificação, provavelmente várias vezes o processo não vai atingir seus requisitos. Para alterar esta situação, dependendo do caso pode ser necessário que a variabilidade seja reduzida, ou que a média do processo seja movida, ou ambos. Uma outra alternativa seria relaxar algum limite de especificação para que os limites do processo e do requisito se alinhem. A figura 3.5 ilustra estas alternativas:

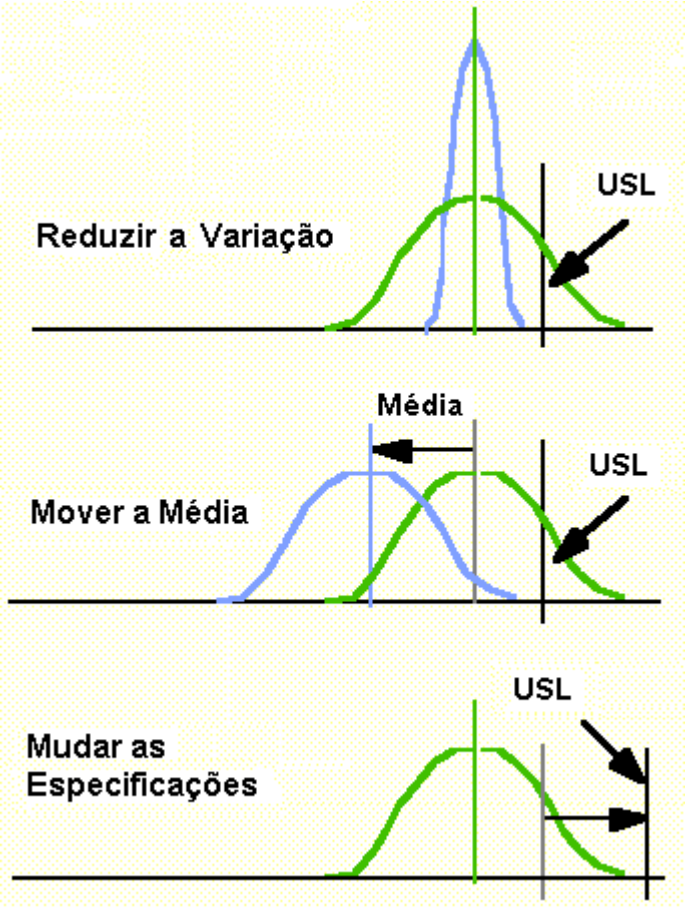

Figura 3-5 - Alinhando o Desempenho do Processo às Especificações do Cliente (Florac et al, 1997, p. 29).

Quando a variabilidade demonstrada pela voz do processo está dentro dos limites especificados pelo cliente, o processo está de acordo com os requisitos do cliente. Quando o processo é estável e se comporta de acordo com os requisitos do cliente, ele é dito *capaz*. Então o conceito de capacidade está ligado tanto à estabilidade do processo quanto à sua habilidade de se comportar de acordo com o requisito do cliente.

3.2.5 MELHORIA DO PROCESSO

A análise do desempenho do processo identifica áreas onde a capacidade do processo pode ser melhorada para melhor suportar os objetivos de negócio. Mesmo processos estáveis podem ter que ser melhorados para satisfazer pressões do mercado ou requisitos do cliente.

Na determinação de melhorias é importante entender não só os objetivos de negócio e estratégias da organização, mas também as prioridades, riscos e problemas associados a estes objetivos e estratégias. A chave neste caso é relacionar os custos e benefícios de cada

proposta de melhoria com os indicadores de negócio usados na organização (Florac, *ibid*, p. 32).

O maior desafio é demonstrar de forma prática e estatística que as mudanças propostas podem ter realmente os efeitos desejados. Quando se está apto a demonstrar que o processo está estável, que são necessárias mudanças para se melhorar o processo, que pode se prever os benefícios da proposta de melhoria do processo baseado em evidências sólidas, a credibilidade e confiança na proposta são melhoradas significativamente. Isto é ainda mais importante quando a proposta requer investimentos (Florac, *ibid*).

3.3 FERRAMENTAS DO SPC

Montgomery (1997, p. 677) define o Controle Estatístico de Processos (SPC) como "uma coleção poderosa de ferramentas de resolução de problemas muito útil para se atingir a estabilidade dos processos e a melhoria da capacidade através da redução da variabilidade".

As principais ferramentas do SPC são os gráficos de controle e análise da capacidade do processo. Outras ferramentas também são usadas para auxiliar na melhoria do processo: Flowcharts, Diagramas de Dispersão, Histogramas, Diagrama de Pareto, Diagrama de Causa e efeito (também conhecido como espinha de peixe ou Ishikawa), Gráficos de tendência e diagramas de afinidade (Gupta, 2002, p. 5).

3.3.1 GRÁFICOS DE CONTROLE

Gráficos de Controle são a ferramenta usada pelo SPC para monitorar e controlar os processos. Segundo Ramos (1997, p.192), os gráficos de controle possuem três objetivos básicos:

1. *"Verificar se o processo estudado é estatisticamente estável, ou seja, se não há a presença de causas especiais de variação.*
2. *Verificar se o processo estudado permanece estável, indicando quando é necessário atuar sobre ele.*
3. *Permitir o aprimoramento contínuo do processo, mediante a redução de sua variabilidade."*

Em geral, gráficos de controle são usados da seguinte forma: são levantados dados referentes ao processo, baseados nos dados são calculados a média e o desvio padrão da amostra, também chamado de Sigma. O gráfico então é criado, mostrando os dados do

processo de forma cronológica, mostrando a linha central, que é a média da amostra, e mostrando os limites do processo, localizados a três desvios padrões (ou três sigmas) da média. Estes limites do processo, também conhecidos como Limites de Controle, são limites sob os quais o processo opera em condições normais. No gráfico são mostrados a **Linha Central** (*CL-Centerline*), que é a média da amostra, o **Limite Superior de Controle** (*UCL - Upper Control Limit*), localizado a três sigmas acima da média e o **Limite Inferior de Controle** (*LCL – Lower Control Limit*), localizado a três sigmas abaixo da média, conforme demonstrado na figura 3.6. O limite superior e o inferior mostram quão longe da média se pode esperar que um valor esteja, considerando a variabilidade normal do processo. O SPC chama esta distância de *magnitude das variações devido a causas comuns* (Leavengood, Reeb, 1999, p.6).

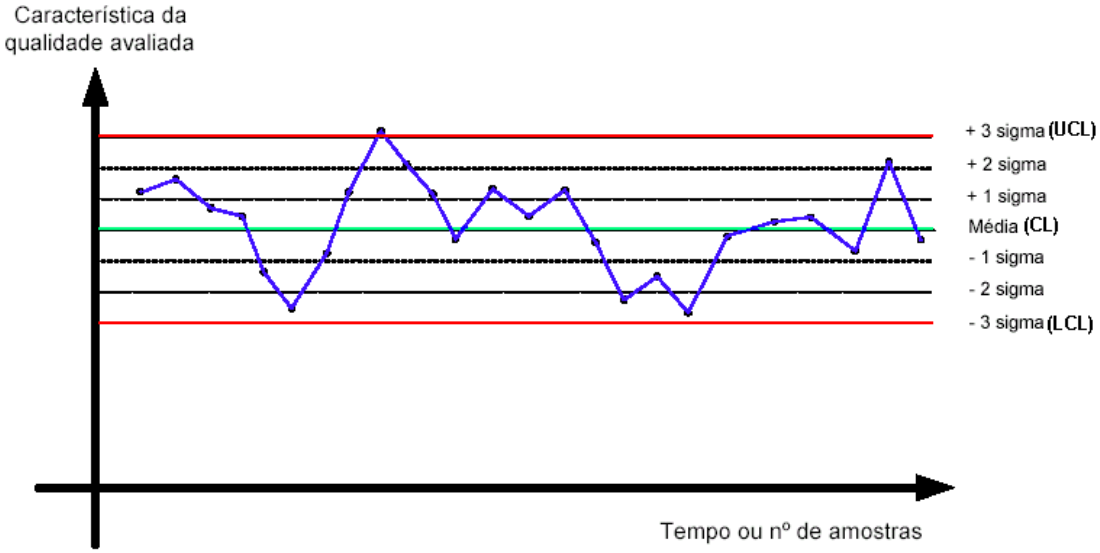

Figura 3-6- Exemplo de gráfico de controle genérico

A utilização dos três sigmas para se definir os limites superiores e inferiores do controle do processo se explica por causa da regra empírica identificada por Wheeler (1992, *apud* Florac et al, 1997, p.160). A regra diz que, dado um conjunto homogêneo (estável) de dados:

- Entre 60 e 70 por cento dos dados se localizarão a uma distância de até 1 sigma dos dois lados da média;
- Entre 90 a 98 por cento dos dados se localizarão a uma distância de até 2 sigmas dos dois lados da média;
- Entre 99 e 100 por cento dos dados se localizarão a até 3 sigmas de distância da média (tanto acima quanto abaixo da média).

Desta forma, é possível perceber que se um dado se encontra acima de 3 sigmas de distância da média, é bastante provável que isso seja devido a uma causa não inerente ao processo.

Os gráficos de controle são um método estatístico para distinguir as variações causadas por "causas comuns" das causadas pelas "causas especiais".

Quando o processo é estável (não existem causas especiais no processo), todos os pontos estarão localizados entre os dois limites de controle, o desempenho do processo é caracterizado por um padrão estável e consistente de valores no tempo (Gupta, 2002, p.6).

Para detectar padrões que demonstram que um processo está fora de controle, os seguintes testes devem ser feitos (Gupta, *ibid*, p.13):

- Teste 1: Um ponto fora dos limites de controle;
- Teste 2: Ao menos 2 de 3 pontos consecutivos do mesmo lado da média e na região entre 2 e 3 sigmas.
- Teste 3: Ao menos 4 de 5 pontos consecutivos no mesmo lado da média, afastados da média por mais de um sigma.
- Teste 4: Ao menos 8 pontos consecutivos do mesmo lado da média.

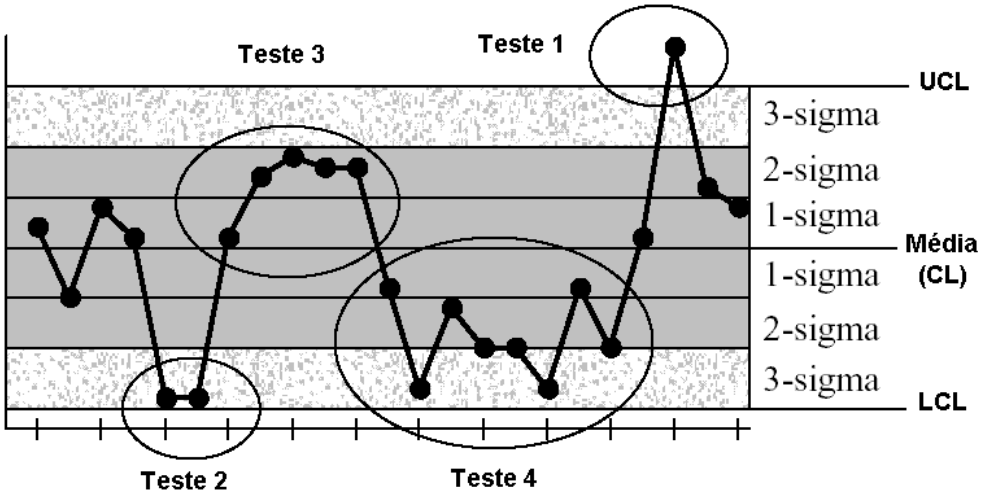

Figura 3-7 - Testes para detectar processos fora de controle (Gupta, ibid, p.13)

Segundo Gupta (*ibid*), existem vários outros testes sugeridos na literatura para detectar padrões que indicariam que um processo esteja fora de controle, que tornam os gráficos de controle mais sensíveis aos sinais, mas ao mesmo tempo podem levar a alarmes falsos.

3.3.2 HISTOGRAMAS

Histogramas usam dados medidos e demonstram a distribuição dos valores observados. Eles são criados agrupando-se os resultados das medidas em células e então contando o número de dados em cada célula. As células não se sobrepõem, têm intervalos de mesma largura em uma escala contínua. A altura de cada barra do histograma é proporcional ao número de ocorrências em cada célula (Florac et al, 1997, p.138).

O histograma no SPC é a ferramenta usada para se avaliar se o processo é capaz, ou seja, se a capacidade do processo está dentro dos limites especificados pelo cliente.

Um cuidado que se deve ter antes de usar o histograma para avaliar a capacidade do processo é garantir que os dados estejam sob controle, ou seja, um gráfico de controle dos dados sempre deve ser utilizado para garantir que o processo esteja sob controle antes de se gerar o histograma.

Ao se traçar o histograma para se calcular a capacidade do processo, é importante que se trace os limites de controle do processo para caracterizar a "voz do processo".

O segundo estágio na análise da capacidade do processo consiste em comparar os limites de controle do processo, a "voz do processo", com os limites impostos pelo cliente, os limites de especificação, ou "a voz do cliente". Da mesma forma que os limites de controle do processo, no caso do limite de especificação é definido o Limite Superior de Especificação (*USL – Upper Specification Limit*) e Limite Inferior de *Especificação (LSL – Lower Specification Limit*). Quando os limites de controle do processo se encontram dentro dos limites de especificação, quase todos os produtos produzidos pelo processo estarão de acordo com as especificações (Florac, *ibid*, p. 110).

Uma forma simplificada de verificar se um processo é capaz ou não é traçar os limites de especificação no histograma com medidas obtidas de um processo estável. O relacionamento entre os limites de controle naturais do processo com os limites especificados pelo cliente irá demonstrar a capacidade do processo (Florac, *ibid*, p. 110).

Na figura 3.8, por exemplo, nota-se que para se atingir os limites especificados pelo cliente o processo deveria sofrer uma melhoria cujo foco seria mover a média (e o processo como um todo) para a esquerda.

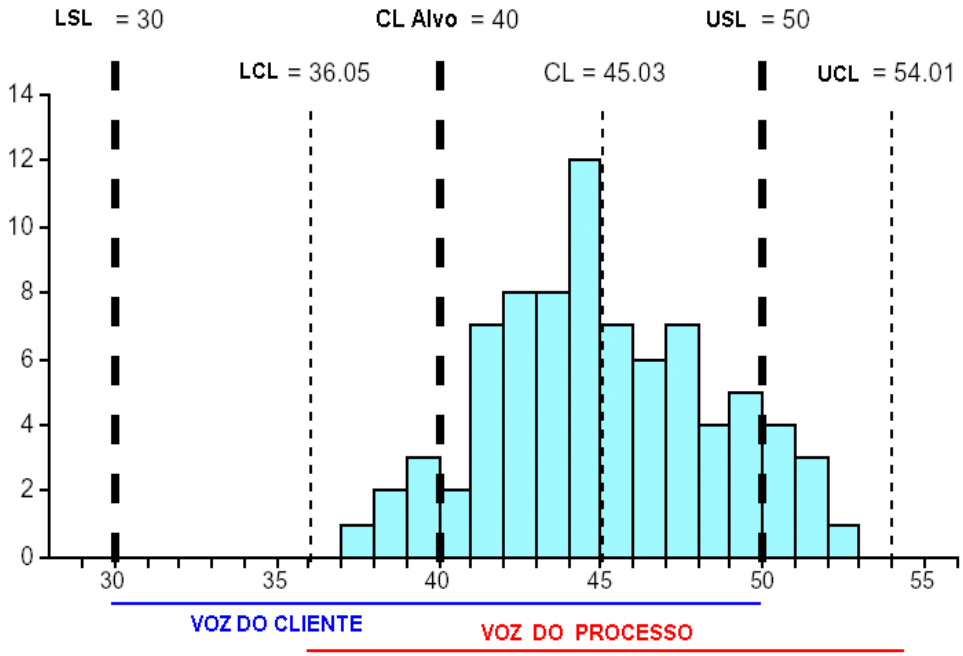

Figura 3-8- Histograma da Capacidade do Processo com Limites de Especificação

3.3.3 DIAGRAMAS DE DISPERSÃO

O diagrama de dispersão (figura 3.9) é um gráfico que demonstra como uma variável se comporta em relação a outra. Geralmente ele é usado como primeiro passo no entendimento dos dados, principalmente na busca de relações de causa e efeito, pois ele demonstra de maneira empírica relações entre duas características de um processo. Um padrão nos pontos traçados pode sugerir que os dois fatores estão correlacionados, talvez em um relacionamento de causa e efeito (Florac, *ibid*, p.129).

Quando um diagrama de dispersão sugere que pode existir algum relacionamento entre duas variáveis, seu uso é freqüentemente seguido pelo uso de uma ferramenta estatística mais formal, como a análise de regressão (Florac, *ibid*, p.131).

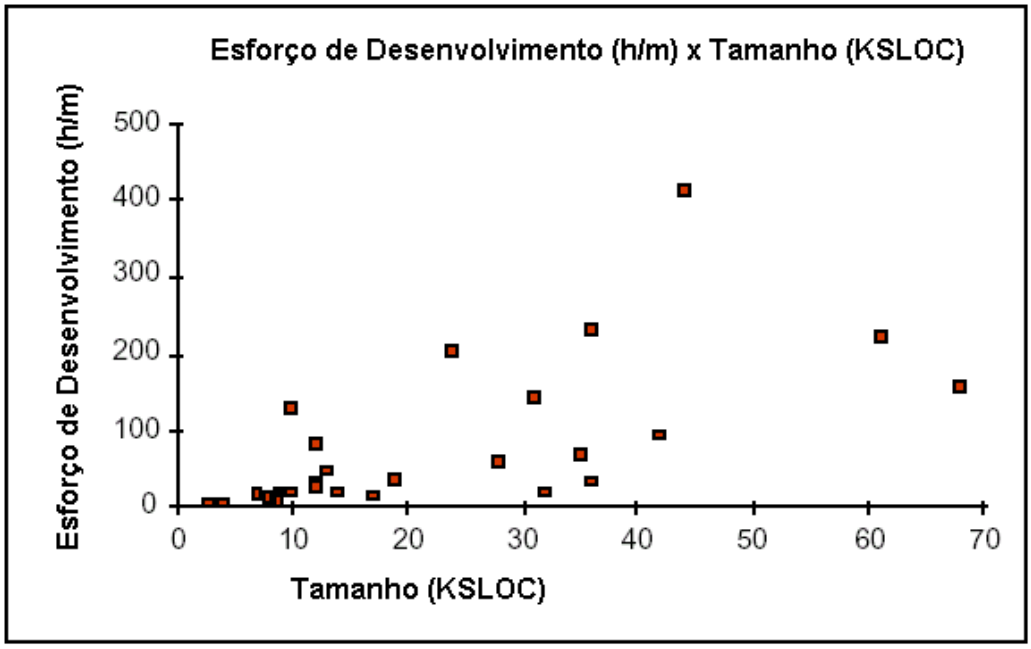

Figura 3-9- Exemplo de Diagrama de Dispersão (Florac, ibid*, p.129)*

Na análise de regressão, é calculado o coeficiente de correlação entre as variáveis. O coeficiente de correlação mais usado é o *R-Pearson*, ou correlação linear. Os valores do coeficiente de correlação variam de –1.00 a + 1.00. O valor –1.00 representa uma *correlação negativa perfeita*, enquanto que o valor +1.00 representa uma correlação positiva perfeita. O valor 0 (zero) representa a ausência de correlação.

Outro valor calculado ao se fazer a análise de regressão é a significância estatística do resultado, ou *P-value*. O *P-value* é uma medida estimada do grau de "verdade" do resultado. Em termos técnicos, o valor do *P-value* representa um índice decrescente de confiabilidade no resultado (Brownlee, 1960): quanto mais alto o *P-value*, menos se pode acreditar que a relação observada entre as variáveis da amostra é um indicador confiável da relação entre as respectivas variáveis na população. Mais especificamente, o *P-value* representa a probabilidade de erro envolvido em aceitar o resultado da correlação (o R – coeficiente de correlação) como válido, ou seja, representativo da população.

Por exemplo: um *P-value* de 0.05 (1/20) indica que existe apenas 5% de probabilidade que a relação encontrada entre as variáveis não seja apenas coincidência. Assumindo, por exemplo, que a correlação (R) encontrada entre as variáveis foi 0.6, pode-se esperar que a cada 100 repetições deste experimento apenas em 5 casos a correlação será menor que 0.6.

Em várias áreas de aplicação, considera-se que um *P-value* de 0.05 é o limite máximo aceitável de erro no processo (representando uma probabilidade de 95% de acerto).

Aplicações não tão rígidas aceitam um *P-value* superior, como 0.10 (probabilidade de 90% de acerto).

3.4 APLICAÇÃO DO SPC PARA O GERENCIAMENTO QUANTITATIVO

Rajesh Gupta (2002,p.8-13), que participou da implantação do SPC na Satyam Computer Services Ltd, Índia, uma organização nível V do CMM, esclarece alguns aspectos na implementação do SPC em uma organização de software(Gupta, 2002):

- **Seleção do processo**: Ele não deve ser usado para analisar todos os problemas de processo na organização. É importante identificar quais são os processos críticos que ajudarão a atingir os objetivos do projeto e da organização (Gupta, *ibid*, p.8-9).
- **Seleção do nível de processo** – A capacidade do processo pode ser calculada teoricamente em qualquer nível: para a organização inteira ou no nível de um projeto, time ou indivíduo. Todavia, a aplicação em um nível muito baixo do processo adiciona pouco valor; a um nível muito elevado se perde o significado. Em um nível muito detalhado do processo, a melhora da capacidade neste nível não necessariamente significará uma melhora no nível do projeto, por exemplo. Por outro lado, em um nível muito alto a variabilidade do processo será tão grande que não provê pistas sobre o processo. Neste caso os limites de controle estarão tão afastados que o processo, mesmo oficialmente sob controle, na verdade não será previsível. Segundo Gupta, na maioria dos casos verificou-se que apenas no nível do sub-processo foi possível gerenciar o processo e implantar melhorias, porque no nível do processo a grande variabilidade dificulta a identificação de possíveis ações de melhoria.
- **Lidando com processos não estáveis** – Sem estabilidade e o conhecimento sobre o desempenho do processo, não é possível distinguir sinal de barulho. Não existe base para reconhecer causas especiais. Neste caso o processo deve ser medido por um tempo suficientemente longo para que seja possível detectar comportamentos não usuais, antes que as técnicas do SPC sejam aplicadas ao processo.
- **Problemas associados a dados** – Vários problemas podem ocorrer ao se iniciar a implantação do SPC:
    - Dados faltantes,
    - Dados suspeitos para alguns atributos,

- Dados imprecisos devido ao excesso de arredondamento,
- Dados muito agregados, que levam a uma grande variabilidade dos limites de controle,
- Dados coletados apenas no final da fase, ou em períodos mensais, muito tarde para se poder manter em controle o processo,
- Muito volume de dados em alguns sub-processos, sendo necessário categoriza-los,
- Quantidade de dados insuficiente devido a poucas interações.

No caso de dados faltantes ou suspeitos, o que se recomenda é que se melhore a definição operacional das medidas, para se obter mais consistência nos dados.

Se não há possibilidade de se melhorar os dados, ou detalha-los para se obter mais dados, o que se aconselha é que não se deixe de começar a aplicar as técnicas do SPC, ao menos gerando o gráfico de controle e os limites de controle iniciais, para uma prévia aplicação de SPC, e que se atualize este gráfico quando houver mais dados. Neste caso a interpretação deve ser mais cautelosa, pois limites de controle calculados com menos de 20 pontos podem não ser confiáveis (Florac, 1997, p.89).

No caso de pouca granularidade de dados, Gupta aconselha a que, antes de interpretar o gráfico de controle, se verifique se os pontos variam ao menos em 5 diferentes níveis de pontos. Pode-se aumentar o número de níveis medindo-se mais precisamente, diminuindo a freqüência das medições para que haja mais variação entre os valores medidos, ou aumentando-se o tamanho dos subgrupos para permitir que mais variações ocorram dentro do subgrupo.

- **Estabilização informal** – Algumas vezes o processo não é consistentemente implementado ou medido como requerido. Isto acontece principalmente quando os dados são coletados de diferentes fontes ou com estratificações diferentes. Neste caso o processo pode ser "informalmente estabilizado" removendo-se os dados que tenham esta causa especial conhecida. Neste caso o dado é simplesmente examinado antes de se criar o gráfico de controle.

3.5 CONCLUSÃO DO CAPÍTULO

Para melhor conhecimento do tema Controle Estatístico de Processos, ou SPC, este capítulo tratou dos seus principais aspectos, por meio da exposição de sua origem, seus principais conceitos, como a variabilidade inerente aos processos, como determinar se um processo está estável ou não e a capacidade do processo.

Também foram apresentadas as principais ferramentas utilizadas no SPC, como o diagrama de controle e o histograma, e como interpretar seus resultados de acordo com os conceitos do SPC.

Ainda neste capítulo foram abordados alguns aspectos e problemas que foram enfrentados por uma organização ao iniciar a implantação do controle estatístico de processos.

Através dessa base teórica, é possível perceber a necessidade de que as medidas a serem utilizadas neste processo estejam em um nível de detalhe adequado, para que se possam descobrir causas comuns de variação, atuar em cima delas para a implementação de melhorias no processo de desenvolvimento de software.

Com a apresentação do CMM e a introdução ao Controle Estatístico de Processos, foi situado o contexto do trabalho e a importância das medidas. O próximo capítulo estará levantando um conjunto básico de medidas do processo de desenvolvimento de software que esteja em um nível de detalhe suficiente para que se possa implementar o controle estatístico de processos em um projeto ou organização, e desta forma se atinja o nível IV do CMM.

4 DEFINIÇÃO DE UM CONJUNTO DE MEDIDAS DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Como definido nos objetivos, as premissas básicas levadas em consideração neste trabalho são:

- É possível identificar um conjunto básico de medidas do processo de desenvolvimento de software que possa ser implementado em quaisquer organizações;
- Se pode encontrar no próprio modelo CMM os elementos essenciais que devem fazer parte deste conjunto.

O artigo que define as práticas chaves do CMM - *Key Practices of the Capability Maturity Model, Version 1.1* (Paulk et al, 1993) propõe várias medidas em cada KPA, basicamente utilizadas para avaliar o status da implementação da KPA. Todavia, em vários momentos o CMM cita também medidas e entidades relacionadas ao processo de desenvolvimento de software. Neste processo de análise será percorrido este artigo buscando identificar estes itens relevantes, que posteriormente terão seus atributos definidos.

As KPAs que relacionam medidas sobre o processo de desenvolvimento de software são aquelas relacionadas ao processo de engenharia de software. O próprio CMM indica quais são estas KPAS (Paulk et al, ibid, p. O-25). Como demonstrado na figura 4.1, as *KPAs* relacionadas aos processos de engenharia de software são:

- Engenharia de Produtos,
- Revisões por Pares,
- Gerência da Qualidade de Software,
- Prevenção de Defeitos.

Categorias de Processos Níveis	Gerenciamento Planejamento do projeto de software, gerenciamento, etc.	Organizacional Revisões pelo gerente responsável, etc.	Engenharia de Software Análise de requerimentos, codificação, teste, etc.
5 - Otimizado		Gerência de mudanças tecnológicas Gerência de mudanças no processo	Prevenção de defeitos
4 - Gerenciado	Gerência quantitativa de processos		Gerência da qualidade de software
3 - Definido	Administração de software integrada Coordenação entre grupos		Engenharia de produtos Revisões
2 - Repetitível	Gerência de requisitos Planejamento de Projetos Acompanhamento de projetos Gerência de subcontratos Garantia de qualidade Gerência de configuração		
1 - Inicial	processos "ad hoc"		

Figure 4-1- KPAs associadas a Categorias de Processos (Paulk et al, ibid, p. O-25)

Neste processo de análise cada uma destas KPAs será avaliada para que se identifiquem itens relevantes a serem incluídos no conjunto básico de medidas do processo de desenvolvimento de software.

4.1 LEVANTAMENTO DAS MEDIDAS BÁSICAS NO CMM

Antes de entrar em cada uma das KPAs, algumas definições da estrutura organizacional do CMM também auxiliam na definição do conjunto. O CMM define o que é Organização, Projeto e Grupo (Paulk at al, *ibid*, p. O-70). Como a intenção do conjunto de medidas é ser implementado em uma organização, não há necessidade de se detalhar a organização como um item do modelo, mas apenas **Projeto** (1) – empreendimento que requer esforço, visando o desenvolvimento ou manutenção de um produto específico, e Grupo –

conjunto de indivíduos que é responsável por um conjunto de tarefas no projeto (Paulk at al, *ibid*, p. O-70). Neste contexto Grupo será considerado sinônimo de **Recursos alocados ao projeto** (2).

4.1.1 KPA DE ENGENHARIA DE PRODUTO

No primeiro item da *common feature* "Habilidade para realizar" da KPA de Engenharia de Produto é exigido que recursos adequados sejam disponibilizados para executar as diferentes fases da engenharia de produto (Paulk et al, *ibid*, p.L3-61). Desta forma é possível identificar que **Recursos** (2) que executam as fases da engenharia de software são elementos importantes do processo. Para medir a adequação dos recursos é importante armazenar o **Nível** (3) de conhecimento que eles têm para executar suas tarefas. O armazenamento do nível do recurso também é uma forma de se permitir análise do desempenho dos recursos indiretamente, através de categorias (níveis), ao invés de se analisar o desempenho de cada indivíduo, que vai contra os princípios do CMM de proteção a dados sobre desempenho individual (Paulk at al, *ibid*, p. L4-3).

Na *common feature* "Atividades realizadas" da KPA de Engenharia de Produto, a segunda atividade mencionada é o levantamento de **Requisitos** de software (4). Como todas as outras fases da engenharia de software dependem dos requisitos, e de forma geral o ciclo de vida do desenvolvimento de um requisito pode ser independente do ciclo de vida de outros requisitos (como em projetos de manutenção, por exemplo), é possível perceber que requisito deve ser mapeado não apenas como um produto de uma fase da engenharia de software, mas como base sob a qual serão planejadas as fases e tarefas necessárias ao completo desenvolvimento do mesmo. Algo que confirma a importância deste elemento é a atividade 10.3, que menciona a necessidade de se ter uma documentação que permita ligar o requisito a todas as fases do ciclo de vida (Paulk et al, *ibid*, p. L3-79).

A partir da atividade 3 são mencionadas as demais fases do ciclo de vida do requisito: desenho de software (atividade 3), codificação (atividade 4), teste (atividade 5), teste integrado (atividade 6), teste de sistema e de aceitação (atividade 7) e documentação (atividade 8) (Paulk et al, *ibid*, p. L3-69-77). Como em uma organização o ciclo de vida de um requisito pode se seguir estas fases de forma mais ou menos detalhadas, ou mesmo podem existir outras fases não previstas no modelo, optou-se por não definir de forma separada cada uma das fases do ciclo de vida de software, mas sim definir o elemento **Fase**

(5), que pode ser customizado para cada organização de acordo com seu ciclo de vida. Segundo o CMM, fases são um conjunto de atividades relacionadas que são realizadas no projeto de software, geralmente vistas como subdivisões do ciclo de vida do projeto de software, e que geralmente terminam com uma revisão formal antes de se prosseguir para a próxima fase (Paulk et al, *ibid*, p. O-58). Neste mesmo lugar é mencionado um outro conceito, o de **Tarefa** (6), definida como uma unidade de trabalho bem definida no processo de desenvolvimento de software, que provê ao gerente do projeto um *checkpoint* visível sobre o andamento do projeto (Paulk et al, *ibid*, p. O-59).

Na atividade 9 é definido que os defeitos identificados em revisões ou em testes devem ser coletados e analisados (Paulk et al, *ibid*, p. L3-78). Então claramente **Defeitos** (7) são elementos importantes a serem incluídos em um conjunto de medidas do processo de desenvolvimento de software.

Na atividade 10 é introduzido o elemento **Mudança** (8) nos requisitos alocados (Paulk et al, *ibid*, p. L3-79). Quando o CMM descreve as medições necessárias à KPA, são mencionados também alguns atributos das mudanças, como categoria da mudança, tamanho, esforço estimado e real da implementação das mudanças (Paulk et al, *ibid*, p. L3-81).

Ainda na KPA de Engenharia de Produto, na *common feature* de "Medição e Análise", o modelo detalha algumas medidas importantes relacionadas a elementos mencionados anteriormente (Paulk et al, *ibid*, p. L3-80), que podem ajudar na definição dos atributos destes elementos:

1. Medidas relacionadas à qualidade – número, tipo e severidade dos defeitos, por fase;
2. Medidas relacionadas à funcionalidade – requisitos por categoria.

4.1.2 KPA DE REVISÃO POR PARES

Na *common feature* "Atividades realizadas" da KPA de *Revisão por Pares*, a terceira atividade os seguintes elementos aparecem como exemplos de medidas a serem coletadas (Paulk et al, *ibid*, p. L3-103):

- identificação do produto sendo revisto;
- tamanho do produto sendo revisto;

- quantidade de participantes da revisão;
- esforço de preparação para a reunião de revisão por revisor;
- esforço gasto na reunião de revisão;
- tipo e número de defeitos encontrados e consertados;
- esforço de retrabalho.

Com esses dados, é possível identificar que o **Produto** (9) é um elemento importante a ser armazenado, com seus atributos de identificação e tamanho. No contexto do CMM, o conceito de Produto é o mesmo de Florac, ou seja, produto de software é qualquer artefato produzido como parte do processo de desenvolvimento de software, como por exemplo documentos de desenho, especificação de programas e software (Florac, 1992, p.7).

Os demais dados mencionados são relacionados a tarefas dentro de fases, como a tarefa de preparo para revisão, revisão e retrabalho.

4.1.3 KPA DE GERÊNCIA DA QUALIDADE DO SOFTWARE

As medidas mencionadas por esta KPA já haviam sido mencionadas nas KPAs de Engenharia de Produto e Revisão por Pares.

4.1.4 KPA DE PREVENÇÃO DE DEFEITOS

Em adição às características de defeitos mencionadas na KPA de Revisão por Pares, a KPA de Prevenção de Defeitos detalha mais características que devem ser armazenadas sobre defeitos. Na *common feature* "Atividades realizadas", atividade 3.4, é determinado que a causa raiz do defeito seja categorizada, como por exemplo em treinamento inadequado, problemas de comunicação, não prestar atenção em todos os detalhes do problema e erros em procedimentos manuais (como erros de digitação) (Paulk et al, *ibid*, p. L5-8). Também na atividade 5 desta mesma KPA são mencionados mais atributos a serem armazenados sobre defeitos (Paulk et al, *ibid*, p. L5-11) :

- descrição do defeito,
- descrição da causa do defeito,
- categoria da causa do defeito (causa raiz, já mencionada),
- fase na qual o defeito foi injetado (fase de origem),

- fase na qual o defeito foi detectado.

Para defeitos identificados como mais críticos, nesta mesma atividade o modelo define ainda mais um conjunto de atributos a serem mantidos como fruto de reuniões de análise de causa raiz, a saber:

- descrição da ação corretiva proposta;
- categoria da ação corretiva proposta.

4.1.5 SUMÁRIO DO LEVANTAMENTO

Após a análise das KPAs relacionadas à engenharia de software, os seguintes elementos foram identificados para fazerem parte do conjunto básico de medidas do processo de desenvolvimento de software:

1) Projeto
2) Recurso,
3) Nível,
4) Requisito (com categoria dos requisitos),
5) Fase,
6) Tarefa,
7) Defeito (com número, tipo, severidade, fase de origem, descrição do defeito, causa raiz, categoria da causa raiz, fase de detecção, e, para defeitos críticos, descrição e categoria da ação corretiva proposta),
8) Mudança (com categoria, custo estimado e real, tamanho estimado e real),
9) Produto (com identificação e tamanho).

4.2 DETALHAMENTO DOS ELEMENTOS DO CONJUNTO DE MEDIDAS

Uma vez identificados os elementos que devem fazer parte deste conjunto básico de medidas do processo de desenvolvimento de software, é importante detalhar seus conceitos e atributos, tanto em termos de domínio quanto de possíveis conjuntos de valores, quando aplicável. Para isso serão procurados no CMM e na literatura atributos e domínios relativos a eles geralmente utilizados para estes elementos. Estes atributos serão documentados

definindo-se uma "Tabela de Conceito/Atributo/Valor" para cada elemento (Ares, Pazos, 1998, p.7). Para que fiquem bem claras as definições dos conceitos, quando possível serão também definidas regras de inclusão e de exclusão (Mellor, Shlaer, 1988, p.34).

PROJETO		
Conceito	Empreendimento que requer esforço, visando o desenvolvimento ou manutenção de um produto específico (Paulk at al, *ibid*, p.O-70).	
Regras de inclusão	Estão incluídos dentro desta definição empreendimentos de desenvolvimento de software. Também estarão inseridos neste conceito empreendimentos que visam a manutenção de software, apesar deste tipo de projeto não ser considerado "temporário" – por exemplo contratos de manutenção de longo termo em sistemas em operação.	
Regra de exclusão	Não se enquadram neste conceito iniciativas que não visem o desenvolvimento de software, apesar de poderem estar gerenciados como projetos, como por exemplo um projeto de desenvolvimento organizacional ou de treinamento.	
Atributo	**Definição**	**Valor**
Nome	Nome pelo qual a organização reconhece o projeto.	Texto
Tipo do Projeto	Classificação do projeto para a organização.	Novo Desenvolvimento ou Manutenção
Plataforma	Ambiente de desenvolvimento do projeto.	Cliente/servidor, *mainframe*, etc.
Data Início	Data em que o projeto iniciou.	Data
Data Encerramento	Data real na qual o projeto foi encerrado.	Data

Tabela 4-1 - Conceito/Atributos/Valores de Projeto

RECURSO		
Conceito	Pessoas com a responsabilidade por alguma tarefa do projeto (Paulk at al, *ibid*, p. O-70).	
Regras de inclusão:	Inclui todo o time de desenvolvimento: analistas, programadores, equipes de teste, equipe de controle de qualidade, documentação, supervisão e gerenciamento (Putnam, 1992, p.363).	
Atributo	**Definição**	**Valor**
Nome do Recurso	Nome pelo qual o recurso é conhecido.	Texto
Data Inicial	Data em que o recurso foi disponibilizado para o projeto.	Data
Data Final	Data em que se encerrou a participação do recurso no projeto.	Data

Tabela 4-2 - Conceito/Atributos/Valores de Recurso

NÍVEL		
Conceito	Nível de experiência do recurso em um certo momento no tempo	
Atributo	**Definição**	**Valor**
Tipo de Nível	Tipos de classificações que a organização utilize para identificar a experiência do recurso	Ex: Anos de Experiência, Cargo, anos de projeto.
Nível	Classificação do recurso dentro do tipo de nível.	Ex: No tipo de nível Cargo, podem ser usados valores como analista junior, pleno ou senior; No tipo de

		nível Anos de experiência a faixa onde o recurso se encontra, como 1 a 3 anos, 4 a 7 anos, etc...
Data inicial	Data em que o recurso foi classificado neste nível	Data
Data Final	Data em que o recurso passou a ser classificado em outro nível	Data

Tabela 4-3 - Coceito/Atributos/Valores de Nível

REQUISITO		
Conceito	Solicitação de Serviço do cliente por uma nova funcionalidade no sistema ou para um conserto/aprimoramento do software. O requisito pode ter origem em levantamento de requisitos propriamente dito, no caso de um projeto de desenvolvimento, ou em solicitações do cliente para melhorias ou consertos de sistemas em produção, quando o projeto for de manutenção (Florac, 1992, p.8).	
Atributo	**Definição**	**Valor**
Nome	Nome resumido pelo qual o requisito é identificado.	Texto
Descrição	Descrição do requisito	Texto
Tipo de Requisito	Classificação do requisito para a organização.	Ex: Correção, novo requisitos, melhoria, adaptação e migração (Putnam, 1992, p.366).

Tabela 4-4 - Conceito/Atributos/Valores de Requisito

FASES		
Conceito	Etapas nas quais o processo de desenvolvimento de software é decomposto para o desenvolvimento do projeto, segundo a metodologia da organização (Paulk et al, *ibid*, p. O-58)..	
Atributo	**Definição**	**Valor**
Nome da Fase	Nome pelo qual a fase é reconhecida dentro da organização.	Ex: Análise, Desenho, Codificação, Teste, Teste de integrado, de sistema e de aceitação, documentação (Paulk et al, *ibid*, p. L3-69-77).
Esforço Estimado	Estimativa do esforço em horas que devem ser dedicadas à fase (Putnam, 1992, p.363).	Numérico.
Duração Estimada	Estimativa de duração mínima, em dias, necessária para se completar a fase com sucesso, levando-se em conta as características de tamanho e produtividade da equipe (Putnam, 1992, p.367).	Numérico.
Esforço Real	Quantidade de esforço em horas gasto para se completar a tarefa.	Numérico.
Duração Real	Diferença em dias entre a data início real e a data fim real.	Numérico.

Tabela 4-5 - Conceito/Atributos/Valores de Fases

A definição das fases de um projeto deve ser realizada em cada organização. Porém, como exposto por Stutzke, em seu trabalho "*Measuring Integrated Product team*" (Stutzke, 2001), não há grande valia em se definir apenas uma única fase. Por outro lado é difícil para

os desenvolvedores distinguirem exatamente onde estão trabalhando quando existem fases demais. É recomendado, então, a utilização das fases utilizadas por Stutzke: análise, desenho, codificação e teste, acrescidas de fases de revisão e retrabalho para cada uma das fases anteriores, pós-implementação, a ser usada ao se cadastrar defeitos identificados após a implementação do requisito.

TAREFA		
Conceito	Unidade de trabalho bem definida dentro de uma fase do ciclo de vida, que permite ao gerente do projeto um *checkpoint* visível sobre o andamento do projeto (Paulk et al, *ibid*, p. O-59).	
Atributo	**Definição**	**Valor**
Nome da Tarefa	Nome pelo qual a tarefa é reconhecida dentro da organização.	Ex: Na fase de desenho: desenho de negócio, especificação de programas, preparo para revisão, revisão e retrabalho (Paulk et al, *ibid*, p. L3-103).
Seqüência	Seqüência da tarefa dentro da fase	Numérico.
Início Estimado	Data em que a tarefa foi planejada para começar.	Data
Fim Estimado	Data em que a tarefa foi planejada para terminar.	Data
Início Real	Data real em que se começou a tarefa.	Data
Fim Real	Data real em que se completou a tarefa.	Data
Esforço Estimado	Estimativa do esforço em horas que deve ser dedicado à tarefa (Putnam, 1992, p.363).	Numérico.
Duração Estimada	Estimativa de duração mínima, em dias, necessária para se completar a tarefa (Putnam, 1992, p.367).	Numérico.
Esforço Real	Quantidade de esforço em horas gasto para se completar a tarefa.	Numérico.
Duração Real	Diferença em dias entre a data início real e a data fim real.	Numérico.

Tabela 4-6 - Conceito/Atributos/Valores de Tarefa

DEFEITO		
Conceito	Defeito de software é qualquer imperfeição em um produto de software ou no processo de desenvolvimento de software (Florac, 1992, p.7).	
Regras de inclusão	Estão incluídos dentro desta definição defeitos identificados tanto em processos de revisão quanto de testes ou na pós-implementação.	
Regras de Exclusão	Não estão incluídos neste conceito eventuais falhas da aplicação após o período de pós implementação.	
Atributo	**Definição**	**Valor**
Número do Defeito	Número seqüencial do defeito	Numérico
Descrição	Descrição textual do defeito.	Texto
Tipo de Defeito	Categorização dos defeitos.	Os tipos de defeitos podem ser defeitos de requisitos, de desenho, de código, de documentação, de casos de teste ou em outro produto do

		ciclo de vida de software (Florac, 1992).
Severidade	Segundo William Florac, Severidade é uma medida do estrago que o defeito causa ao usuário, quando este o encontra.	É geralmente medida em 5 níveis, sendo o mais crítico caracterizado como algo catastrófico e o menos crítico como um nível aceitável (Florac, 1992).
Processo Identificador	Processo que identificou o defeito.	Revisão, teste ou operação. Obs: Em seu relatório, Florac chama o atributo de Processo Identificador de "Finding activity" e propõe 28 classificações, juntando-se a fase e o processo identificador em si (Florac, 1992). Nesta proposta, procura-se manter separadas a fase em que o defeito foi identificado e processo que o identificou.
Fase de Detecção	Fase do ciclo de vida de desenvolvimento no qual o defeito foi identificado[1].	Ex: Análise, Desenho, Codificação, Teste, Teste de integrado, de sistema e de aceitação, documentação (Paulk et al, *ibid*, p. L3-69-77).
Fase de Origem	Fase do ciclo de vida de desenvolvimento onde o defeito foi introduzido[2].	Como acima.
Data de Detecção	Data em que o defeito foi detectado.	Data
Data de Eliminação	Data em que o defeito foi eliminado.	Data
Causa Raiz	Causa principal do defeito	Texto
Categoria da Causa raiz	Categorização da causa raiz, para facilitar futuras pesquisas.	Ex: Treinamento inadequado, problema de comunicação, não atenção a todos os detalhes do problema e erros em procedimentos manuais (Paulk et al, *ibid*, L5-11).
Ação Corretiva	Descrição da ação corretiva que foi definida para atacar a causa raiz do defeito (Paulk et al, *ibid*, L5-11).	Texto
Categoria da Ação Corretiva	Categorização da ação corretiva que foi definida para atacar a causa raiz do defeito (Paulk et al, *ibid*, L5-11).	Texto

Tabela 4-7 - Conceito/Atributos/Valores de Defeito

1 As fases de detecção e de origem devem estar de acordo com as fases definidas para a requisição específica (não teria sentido se fossem identificados defeitos na fase de análise, se esta fase não ocorreu para a requisição).

2 A respeito da fase de origem, analisando-se o número de defeitos identificados em uma fase e os originados nesta mesma fase, pode-se determinar os defeitos que deixaram de ser identificados pelos processos normais de revisão e teste. Em seu trabalho "*Software Quality Measurement: A Framework for Counting Problems and Defects*" (Florac, 1992, p.4), Florac dá o nome a esta diferença de "defeitos que escaparam" dos processos de detecção na fase, que afetaram as fases seguintes

MUDANÇA		
Conceito	Mudança solicitada nos requisitos após os mesmos terem sido já aprovados.	
Atributo	**Definição**	**Valor**
Descrição	Descrição da mudança sendo solicitada	Texto
Tamanho	Tamanho estimado e real da mudança. Tamanho pode ser medido em pontos de função, linhas de código incluídas, alteradas ou excluídas por linguagem, páginas de documentação, etc.	Numérico
Esforço estimado/real	Esforço em horas necessário para implementar a mudança.	Numérico
Duração Estimada/real	Estimativa de duração mínima, em dias, necessária para se completar a mudança (Putnam, 1992, p.367).	Numérico

Tabela 4-8 - Conceito/Atributos/Valores de Mudança

PRODUTO		
Conceito	Qualquer artefato produzido como parte do processo de desenvolvimento de software (Florac, 1992, p.7).	
Regras de inclusão:	Documentos, planilhas e programas gerados pelo ciclo de vida de desenvolvimento do software no atendimento ao requisito.	
Regra de exclusão	Quaisquer artefatos não ligados ao ciclo devida de desenvolvimento de software (ex: controle de bonificações do projeto).	
Atributo	**Definição**	**Valor**
Nome do produto.	Nome pelo qual a organização reconhece o produto.	Texto
Tipo de produto	Classificação do produto.	Ex: software, documento de análise, especificação de programa, plano de teste.
Tecnologia	Tecnologia utilizada para criar/manter o produto	Ex: Word, Excel, Cobol, C++, Cobol.
Tamanho	Tamanho do produto estimado ou entregue em um projeto. Tamanho pode ser medido em pontos de função, linhas de código incluídas, alteradas ou excluídas por linguagem, páginas de documentação,etc.	Numérico

Tabela 4-9 - Conceito/Atributos/Valores de Produto

Para resumir todos os elementos identificados e seus conceitos, eles foram modelados em um diagrama conceitual de dados[3], usando a notação do diagrama de classes da UML (OMG, 2003).

[3] O processo de modelagem está descrito no Apêndice 8.1.

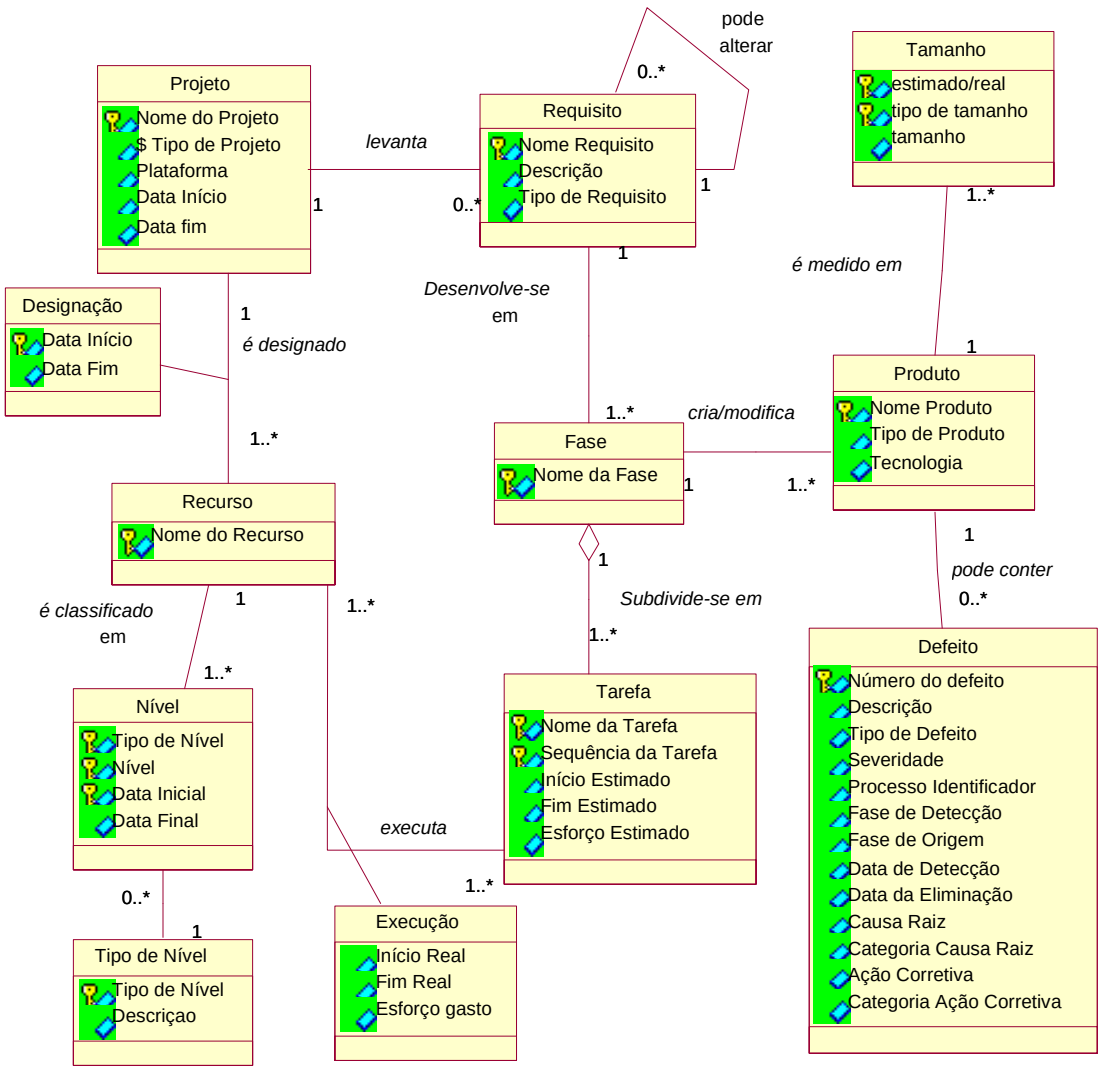

Figura 4-1 – Medidas do Processo de Desenvolvimento de Software

Interpretando o modelo, de acordo com a metodologia adotada na organização, o **Projeto** tem um Grupo de **recursos designados** a si, com a finalidade de implementar os **Requisitos** levantados. Podem ser solicitados **Requisitos** que promovam mudanças em **Requisitos** em andamento. O desenvolvimento dos Requisitos *se estrutura* em **Fases,** que *constróem* diversos **Produtos** derivados do processo de engenharia de software. Cada **Produto** *é medido em* **Tamanhos,** estimado e real, seja este produto um documento, cujo tamanho seria medido em páginas de documentação; ou software, medido em pontos de função e/ou linhas de código, por exemplo. O **Produto** gerado *pode conter* **Defeitos**, que terão uma série de atributos definidos visando identificá-los e classificá-los, de forma que a

organização possa identificar razões em comum para os defeitos e propor melhorias no processo. As **Fases** se subdividem em **Tarefas.** Cada **Tarefa** tem um esforço e duração estimados e, baseado nisto, os **Recursos** disponíveis são colocados à disposição para executarem as tarefas.

4.3 CONCLUSÃO DO CAPÍTULO

Baseado na premissa de que é válido definir um conjunto básico de medidas que possa ser implementado em organizações de software, e que este conjunto poderia ser encontrado no CMM, neste capítulo foi realizado um levantamento das KPAs do CMM relacionadas à engenharia de software. Cada uma delas foram analisadas para que se definisse um conjunto de medidas básicas do processo de desenvolvimento de software.

Definidos os elementos que deveriam fazer parte deste conjunto básico, eles tiveram seus conceitos e atributos definidos, baseando-se no próprio CMM e na literatura relacionada à engenharia de software.

Para fins didáticos, este conjunto foi modelado em um diagrama conceitual, usando-se a notação de classes da UML.

Para analisar-se se é viável implementar este conjunto de medidas em uma organização, e se este conjunto realmente auxilia na implementação do controle quantitativo de processos e na obtenção do nível IV do CMM, foi realizado um estudo de casos, no qual implementou-se este conjunto de medidas em um projeto. Este estudo de casos será descrito no próximo capítulo.

5 ESTUDO DE CASO

O objetivo deste estudo de caso é demonstrar a aplicabilidade do conjunto de medidas do processo de software a uma organização que queira implementar o controle quantivativo de processos. Também visa demonstrar a robustez do conjunto, avaliando-se se a essência do mesmo se manteve durante o estudo de caso.

Para desenvolver um estudo de caso, este conjunto foi utilizado em um projeto de uma organização de software.

O objetivo em vista era que fosse criado um repositório de dados baseado no conjunto básico de medidas proposto neste trabalho e, uma vez estando o repositório preenchido com um número significativo de dados, fosse possível utiliza-lo para controlar estatisticamente os processos relacionados aos objetivos do projeto, bem como fazer inferências estatisticamente significativas, que pudessem indicar pontos de melhoria nos processos. Desta forma seria confirmado se o conjunto proposto é viável e realmente atende aos requisitos do CMM IV.

Torna-se importante salientar que neste estudo de caso foi ressaltada a implementação do conjunto de medidas e sua utilização no processo de gerenciamento quantitativo do projeto selecionado. Várias outras iniciativas e processos foram implementados no nível organizacional para a consolidação de características de maturidade da organização, mas que estão fora do escopo deste trabalho detalhar.

5.1 O PROJETO ESCOLHIDO

O projeto escolhido foi o projeto BSCA, um dos projetos da fábrica de software da EDS Brasil - *Rio Solution Centre* (RioSC). Este tem por objetivo fazer manutenções e melhorias no sistema de *Health Care* corrente da empresa *Blue Shield of California*. A plataforma do sistema é *mainframe*, usando sistema operacional MVS e as linguagens Cobol, Assembler e JCL. Os dados são armazenados em arquivos VSAM.

Esse contrato de manutenção pertence à empresa a mais de 15 anos e, a partir de Outubro de 2001 iniciou-se a participação neste projeto de uma equipe do Brasil. Esta equipe no Brasil, chamada de "Best Shore BSCA" é composta de 16 pessoas e esta participação é administrada como um projeto. Este projeto no Brasil foi o escolhido como

piloto para a implementação do conjunto básico de medidas do processo de desenvolvimento de software.

O projeto no Brasil é responsável pela fase de construção e teste unitário dos requisitos do sistema de *Claims* (sinistros). Em média estas etapas de construção e teste unitário levam 140h. As demais fases do ciclo de vida são realizadas pela organização da empresa nos Estados Unidos: de análise a desenho, depois teste integrado, teste de aceitação do usuário e implementação.

5.2 A UTILIZAÇÃO DO CONJUNTO DE MEDIDAS NO PROJETO BSCA

O primeiro passo para se iniciar a utilização deste conjunto foi a sua intanciação em um repositório de dados. Para isso definiu-se a nomenclatura de campos e valores padrões utilizados pela metodologia da empresa, chamada GSMS (*Global Solution Management System*).

O segundo passo foi mapear os dados pré-existentes neste repositório, de forma a criar uma base histórica[4]. Para explicar como foi feito este mapeamento, para cada elemento do conjunto será descrito de onde os dados foram obtidos e o domínio utilizado no projeto, quando apropriado.

1) Projeto

O itens de dados relacionados ao nome do projeto, plataforma e data inicial foram obtidos no formulário de registro do projeto no banco de dados organizacional da empresa *(Presage Metrics)*.

2) Recurso e Designação

Os dados dos recursos e de sua associação ao projeto foram obtidos a partir do plano de alocação do projeto, armazenado em uma planilha Excel.

3) Nível do Recurso

A partir também do plano de alocação do projeto foi possível obter o nível do recurso. O projeto BSCA utilizava dois tipos de níveis para classificar os recursos: anos de experiência em tecnologia da informação e anos de experiência no sistema.

---

[4] É importante ressaltar que várias medidas aqui levantadas já eram levantadas anteriormente e enviadas para uma base de dados geral da organização – o *Presage Metrics*, que armazena as métricas organizacionais. Estas métricas eram mantidas em um nível de granulosidade suficiente para a organização como um todo, mas não no nível necessário para o projeto, que precisaria de informações detalhadas por requisito para poder melhorar o processo de desenvolvimento dos mesmos.

4) Requisitos e 8) Mudanças

Os dados básicos relativos a requisitos e mudanças foram obtidos de uma aplicação da EDS de gerenciamento de projeto. Nesta aplicação são cadastradas mudanças na aplicação (requisitos de melhorias ou mudanças na aplicação) e mudanças no projeto (mudanças em requisitos em andamento).

Os dados relativos ao esforço e duração dos requisitos e mudanças foram obtidos através do plano do projeto, que estava armazenado em MSProject. Os tipos de requisito já eram classificados, de acordo com a metodologia GSMS, que em um projeto de manutenção utiliza a seguinte classificação dos tipos de requisitos: Melhoria, Adaptação, Correção e Aperfeiçoamento. Estes foram os possíveis tipos de requisitos usados no modelo. Os números e nomes dos requisitos foram armazenados exatamente como encontrados no plano do projeto.

5) Fase e 6) Tarefa

Todos os dados relativos às fases do projeto foram obtidos através do plano do projeto no *MSProject*. A fase de Produção (única fase executada pelo projeto no Brasil), de acordo com a metodologia GSMS é quebrada nas seguintes tarefas: Codificação, Teste Unitário, Revisão e Retrabalho. Pela própria definição da metodologia, no plano do projeto eram criadas estas tarefas para cada requisito. Desta forma foi possível obter todos os dados necessários relacionados às fases e tarefas.

Uma vez aprovadas as estimativas elas são planejadas utilizando-se o MSProject e este plano é importado pelo sistema de controle de horas trabalhadas (CTTS). É neste sistema que os recursos registram as horas trabalhadas em cada tarefa do requisito. O sistema mantém as datas e esforço gasto por recurso em cada tarefa dia a dia, utilizando uma base de dados MSAccess, e re-alimenta estas horas e datas no MSProject para acompanhamento. Desta forma foram obtidos os dados relativos à designação dos recursos às tarefas (recurso, requisito, fase, tarefa, data início e fim e esforço real).

7) Defeito

Defeitos provenientes de Revisão - Os defeitos encontrados através da revisão de cada produto do requisito eram armazenados em formulários Word. Estes defeitos eram sumarizados para a coleta de métricas da organização por tipo de defeito, fase de origem e

fase de identificação. Não havia uma coleta de defeitos por requisito. Para popular este elemento do conjunto foi necessário implementar uma macro que coletasse os defeitos por requisito diretamente dos formulários Word. Todos os atributos do elemento Defeito já eram encontrados no formulário.

Defeitos provenientes de teste – O projeto BSCA utilizava uma planilha para a criação e acompanhamento dos casos de teste. Os defeitos identificados na fase de teste já eram catalogados nesta planilha e classificados. Desta forma foram facilmente mapeados no conjunto de medidas.

9) Produto

Os produtos mantidos/desenvolvidos em cada requisito foram obtidos através da matriz de rastreabilidade que era criada para cada requisito. Esta matriz incluía tanto os produtos do tipo software quanto os não-software, como documentos de desenho e documentação de testes. O tipo de produto também foi possível obter através desta matriz. Todavia, a informação sobre tecnologia, no caso dos produtos do tipo software, somente era armazenada no documento de estimativa. Então este documento também foi utilizado para se mapear esta informação.

No projeto BSCA as estimativas de cada requisito e mudanças são feitas e mantidas em planilhas Excel. Nesta estimativa são documentados os programas a serem alterados, com seus tamanhos e percentuais de alteração, e cada fase e tarefa do requisito com seu esforço estimado. A partir desta informação foi possível obter o tamanho estimado em linhas de código.

Uma vez encerrada a tarefa, os recursos da equipe realizam a contagem das linhas de código por linguagem, armazenando estas informações em outra planilha Excel. A partir deste preenchimento foi obtido o tamanho real de cada requisito, em linhas de código.

Todo este mapeamento foi feito para se entender como o conjunto de medidas seria alimentado no projeto BSCA. Uma vez definido o mapeamento, foram criadas macros para que o repositório de dados fosse alimentado automaticamente, de acordo com o ciclo de vida do requisito.

5.3 IMPLEMENTANDO CONTROLE QUANTITATIVO

Uma vez alimentado o repositório de dados que foi construído baseado no conjunto de medidas do processo de desenvolvimento de software, foi possível começar a análise das medidas levantadas visando a implementação do controle quantitativo no projeto.

O projeto BSCA, juntamente com outros projetos do RioSC, foi selecionado para iniciar a implementação do CMM IV. Os recursos do projeto receberam treinamento de recursos da EDS de outros países, cujos projetos já haviam sido certificados neste nível.

Os passos seguidos pelo projeto para a implementação do controle quantitativo foram baseados em metodologia própria da EDS, criada a partir da consolidação de uma série de modelos propostos na literatura. Os pontos principais foram sintetizados neste trabalho para possibilitar uma demonstração da utilização do conjunto de medidas na implementação do controle estatístico dos processos.

5.3.1 DEFININDO OBJETIVOS

O primeiro passo tomado foi definir os objetivos de qualidade e de desempenho do processo baseados nas expectativas do cliente e nos objetivos da organização.

Ao começar a trabalhar com as expectativas do cliente percebeu-se que elas estavam em um nível muito abstrato: suas expectativas eram "Aumento da Eficiência" e "Aumento da Produtividade". Foi tomada a decisão, então, de expandir estas expectativas através do uso do diagrama "CTQ Tree" (*Critical to Quality Tree*) (em Apêndice). A idéia desta ferramenta é que se consiga mover de expressões gerais que são difíceis de medir para requisitos específicos que sejam mais facilmente mensuráveis. Através do uso desta ferramenta foram detalhados os seguintes objetivos:

1. Reduzir a variação entre o esforço estimado e real;
2. Aumentar o percentual de defeitos identificados na fase de Construção,
3. Reduzir o esforço gasto em retrabalho,
4. Manter os níveis de utilização, com os recursos trabalhando em requisitos em média 140h por mês.

Com relação aos objetivos da organização do RioSC, ela já havia definido objetivos gerais que ela gostaria que fossem seguidos para todos os projetos. Destes objetivos o que se aplicava às características do projeto era a redução da variação entre o esforço estimado e o real. Como este objetivo também era um dos objetivos levantados junto ao cliente, o projeto começou seu processo considerando este conjunto de quatro objetivos.

5.3.2 PRIORIZANDO OBJETIVO A SER CONTROLADO QUANTITTATIVAMENTE

Como havia um limitado número de recursos que poderiam atuar na melhoria dos processos, era muito importante identificar quais objetivos teriam maior prioridade. Para isso foi utilizada uma ferramenta que possuía uma forma simples de ordenar os objetivos: são estabelecidos critérios de avaliação, definidos pesos para cada critério e por fim cada objetivo é pontuado contra os critérios estabelecidos. A prioridade é dada selecionando-se os objetivos com maior pontuação.

No caso do projeto BSCA os objetivos foram pontuados contra os critérios "Benefício para o Cliente", "Criticidade" (se atualmente aquele objetivo representava um problema para o cliente), "Possibilidade de implementação" (se o processo relacionado àquele objetivo estava sob controle do projeto no Rio) e "Facilidade de implementação" (se o custo para implementar este objetivo seria alto ou baixo), conforme a figura a seguir.

Objetivos do Projeto	Benefício para o Cliente Nota 1 a 3 Alto = 3 Baixo = 1	Peso	Críticidade Nota 1 a 3 Alto = 3 Baixo = 1	Peso	Possibilidade de Implementação Nota 1 a 3 Alto = 3 Baixo = 1	Peso	Facilidade de Implementação Nota 1 a 3 Fácil = 3 Difícil = 1	Peso	Prioridade Total
Reduzir a variação entre o esforço estimado e real.	3	3	3	4	3	2	2	1	**29**
Aumentar o percentual de defeitos identificados na fase de Construção.	3	3	1	4	2	2	2	1	**19**
Reduzir o esforço gasto em retrabalho.	1	3	2	4	2	2	1	1	**16**
Manter os níveis de utilização, com os recursos trabalhando em requisitos em média 140h por mês.	2	3	1	4	1	2	1	1	**13**

Figura 5-1 Priorização dos Objetivos

Através desta ferramenta foi identificado que o objetivo prioritário era "Reduzir a variação entre o esforço estimado e o real".

O objetivo priorizado seria colocado sob controle estatístico. Os demais objetivos seriam apenas monitorados mensalmente. Para que fosse possível esse acompanhamento era importante que as medidas relacionadas a estes objetivos não priorizados também constassem do conjunto básico de medidas, e isso se mostrou verdadeiro, como demonstrado pela tabela 5.1.

Objetivos não priorizados	Medidas relacionadas	Parte do conjunto básico?
Aumentar o percentual de defeitos identificados na fase de construção	Soma de Defeitos do requisito com fase de detecção = Construção / Total de Defeitos do requisito	Sim
Reduzir o esforço gasto em retrabalho	Esforço gasto na tarefa "Retrabalho"/ Esforço total na Fase	Sim
Manter os níveis de utilização, com os recusos trabalhando em média 140h por mês	Soma de esforço real do recurso no mês (Designação)	Sim

Tabela 5-1 - Medidas relacionadas aos objetivos não priorizados

5.3.3 AVALIANDO A ESTABILIDADE E CAPACIDADE DO PROCESSO

Uma vez definido o objetivo a ser controlado quantitativamente - "Reduzir a variação entre o esforço estimado e o real", era necessário conhecer o comportamento do principal processo afetado por este objetivo, que era o processo de estimativa. Entendendo junto ao cliente e à organização que o que era importante neste objetivo era o percentual de variação e não a variação em horas, foi definida a principal métrica a ser controlada quantitativamente:

Variação de esforço = (esforço real – esforço estimado)/esforço estimado

Como estes dados já faziam parte do conjunto básico de medidas e estavam armazenados no repositório de dados do projeto, foi possível criar o gráfico de controle da variação de esforço em cada requisito[5], demonstrado pela figura 5.2. No momento da medida haviam 22 pontos distribuídos em 3 meses.

[5] Nos gráficos do estudo de caso os valores relativos à variação de esforço foram suprimidos por se tratar de informação classificada. Entende-se que esta supressão de modo algum afeta o endendimento dos gráficos

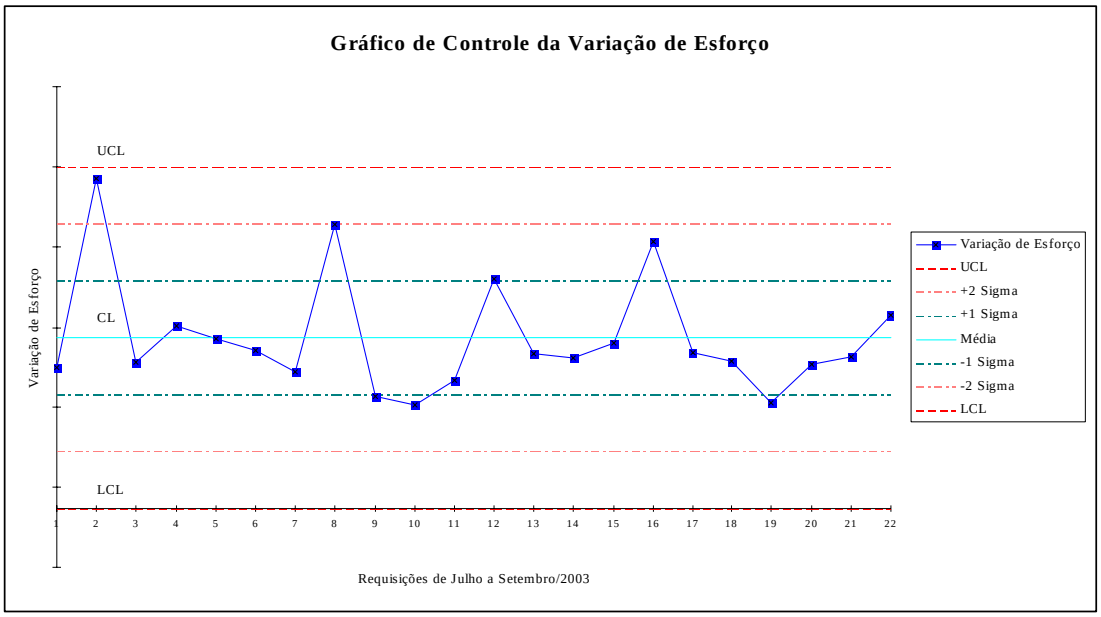

Figura 5-2 - Gráfico de Controle da Variação

Analisando o gráfico constatou-se que o processo estava estável e sob controle, pois o processo não foi reprovado em nenhum dos 4 testes que avaliam se o processo está fora de controle descritos no capítulo de SPC: não haviam pontos fora dos limites de controle, não haviam 2 de 3 pontos consecutivos do mesmo lado da média e na região entre 2, e 3 sigmas, etc..

O passo seguinte foi traçar o histograma, para avaliar se o processo estava capaz do ponto de vista do cliente (figura 5.3).

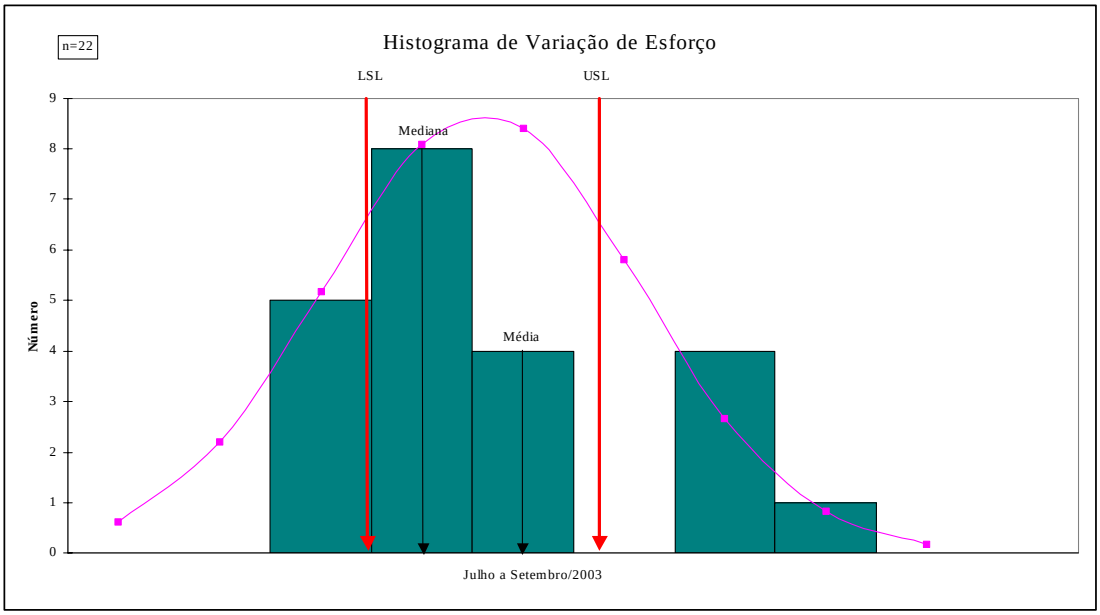

Figura 5-3 - Histograma - Variação de Esforço

Com o resultado foi possível entender que o processo ainda precisava melhorar, pois a variação de esforço em vários casos não estava entre os limites desejados pelo cliente (*Upper Specification Limit* – USL e *Lower Specification Limit*). Então foi decidido detalhar os sub-processos envolvidos no processo de estimativa para melhor compreendê-lo e identificar o sub-processo que mais afetava esta variação.

5.3.4 IDENTIFICANDO OS SUB-PROCESSOS E SUA INFLUÊNCIA

Entendendo o processo de estimativa definido no projeto e também levantando junto aos recursos todos os passos que eles executavam para criar uma estimativa, foi identificado que o processo de estimativa se compunha de seis sub-processos :

1. Identificar complexidade dos módulos afetados (baseado no tamanho e percentual dos módulos afetados);
2. Calcular esforço de construção por modulo;
3. Definir percentual de ajuste baseado no conhecimento do recurso;
4. Calcular esforço de outras fases baseado no esforço de construção;
5. Documentar estimativa (premissas/riscos);
6. Revisar Estimativa.

Para melhor entender o relacionamento entre os sub-processos, bem como suas entradas e saídas, foi desenhado o mapa do processo de estimativa (figura 5.4).

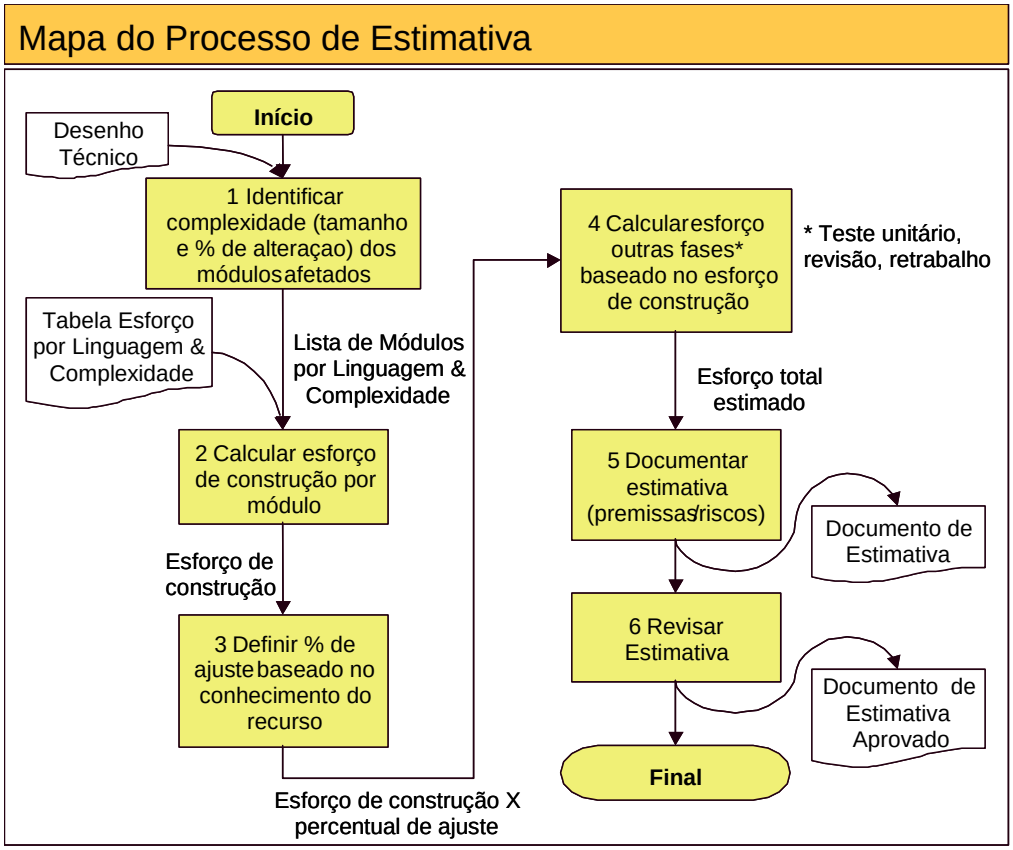

Figura 5-4 - Mapa do Processo de Estimativa

Para compreender ainda melhor os sub-processos envolvidos, foi realizado um *brainstorm* com os recursos do projeto para identificar todas as medidas possíveis de se extrair destes sub-processos. O resultado deste *brainstorm* foi documentado na tabela 5.2.

Sub-processo	**Medidas**
1. Identificar complexidade dos módulos afetados	• Total de módulos/linguagem • Número de módulos novos/linguagem • Número de módulos por tamanho (pequeno, médio, grande)/linguagem • Número de módulos por percentual de alteração (baixo, médio, alto)/linguagem • Número de módulos por complexidade/linguagem • Linhas de código estimadas/linguagem
2. Calcular esforço de construção por módulo	• Esforço por complexidade/linguagem
3. Definir % de ajuste baseado no conhecimento do recurso	• Anos de experiência em tecnologia da informação • Anos de experiência no sistema

4. Calcular esforço das outras fases baseado no esforço de construção	• % de construção/total • % de teste unitário/total • % de revisão/total • % de retrabalho/total
5. Documentar estimativa	• Esforço gasto documentando a estimativa
6. Revisar Estimativa	• Esforço gasto na revisão da estimativa • Esforço de retrabalho na estimativa após a revisão

Tabela 5-2 - Medidas relacionadas aos suprocessos

Para tentar descobrir quais destes sub-processos teriam maior influência na variação de esforço, foi feita uma análise de regressão separadamente de cada uma destas variáveis contra a variação de esforço e também contra a variação não ajustada (considerando o esforço estimado sem o ajuste proveniente do passo 3). O resultado obtido foi não conclusivo, uma vez que todas estes pares de variáveis obtiveram fracas correlações.

Continuando a analisar todas as opções possíveis, no entanto, foi identificado que também era muito baixa a correlação entre a variação de esforço em horas (esforço real – esforço estimado) e o esforço calculado do ajuste, como demonstrado na figura 5.5.

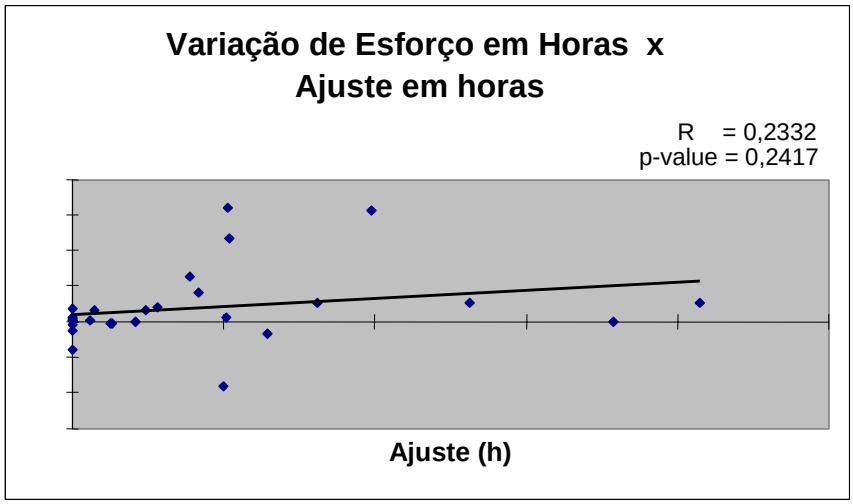

Figura 5-5 - Correlação Variação x Ajuste (h)

Foi decidido então que seria necessário investigar melhor se os fatores que estavam sendo usados para definir o percentual de ajuste realmente tinham influência na variação de esforço. Para isso foi calculada a correlação entre a variação de esforço não ajustada e os fatores que estavam sendo utilizados para fazer o ajuste ("Anos de Experiência em Tecnologia da Informação" e "Anos de Experiência no Sistema") (figura 5.6 e 5.7).

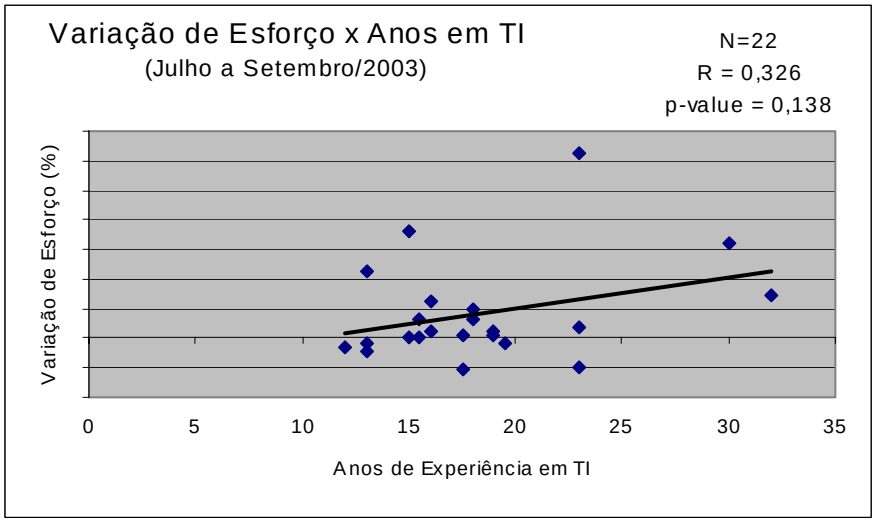

Figura 5-6 - Correlação Variação x Experiência em TI

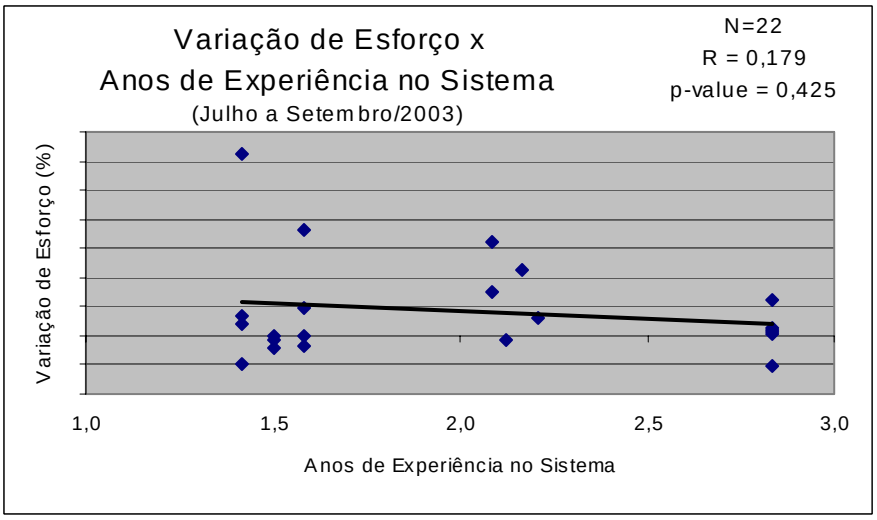

Figura 5-7 - Correlação Variação x Experiência no Sistema

Analisando estes gráficos foi possível perceber que as variáveis que estavam sendo utilizadas como base para o ajuste da estimativa tinham estatisticamente pouca relação com a variação de esforço.

5.3.5 MELHORANDO O PROCESSO

Como todos os outros pares de variáveis também demonstravam ter baixa correlação, foi levantada a hipótese de que talvez outra medida, ainda não levantada, pudesse estar

exercendo uma influência maior que as anteriores. Para tentar descobrir que medida seria esta foi realizado com o grupo um levantamento de todas as possíveis causas para a variação, utilizando o diagrama de causa e efeito (ISHIKAWA).

Analisando as causas repetidas em todas as categorias do diagrama de causa e efeito, chegou-se à conclusão que o nível de conhecimento/proficiência que os recursos tinham no subsistema relacionado ao requisito provavelmente poderia ter influência na variação de esforço.

Estas duas informações – proficiência do recurso por subsistema e subsistema de cada requisito – não eram medidas que constavam do conjunto básico de medidas proposto. Também não eram medidas que já estavam modeladas de alguma forma no projeto. Era necessário criar o processo que definia a proficiência do recurso por subsistema e o processo que identificava a qual subsistema um certo requisito pertencia.

Para medir o nível de proficiência dos recursos em cada subsistema foi criado um questionário com diversos itens a serem avaliados em cada subsistema. Para cada item o recurso deveria avaliar seu nível de conhecimento em quatro níveis de proficiência:

1. Iniciante
2. Conhecimento básico
3. Proficiente
4. Especialista

O nível de proficiência do recurso no subsistema era definido como o nível de proficiência mais vezes utilizado como resposta. Todos os recursos se submeteram a este questionário e o conhecimento de cada um nos subsistemas foi avaliado (figura 5.8).

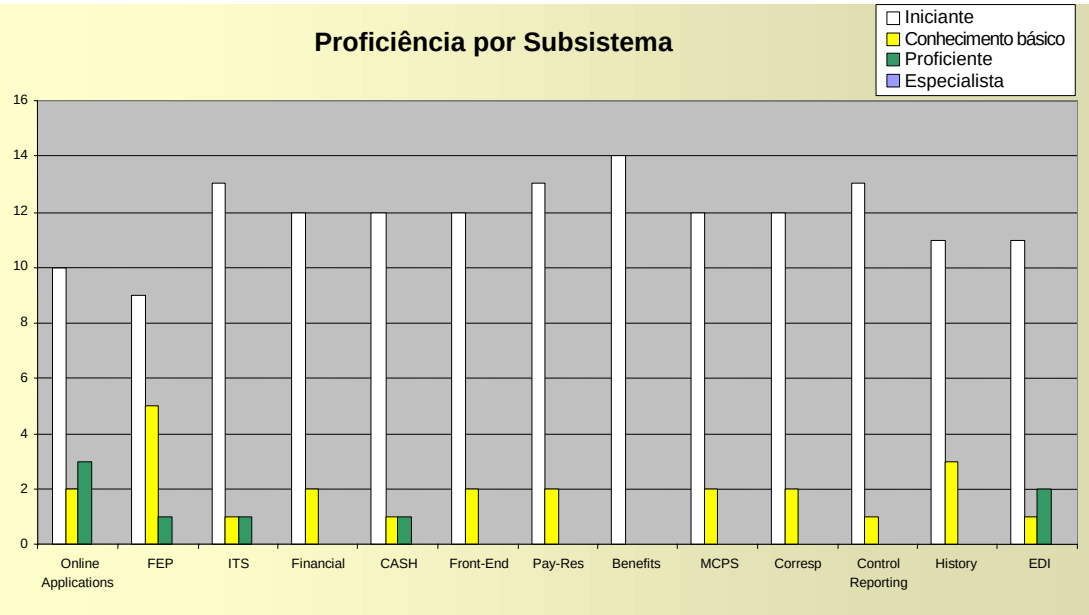

Figura 5-8 - Nível de Proficiência por Subsistema

Desta forma foi estabelecido o processo de definição da proficiência do recurso por subsistema. No processo foi detalhado que a cada 6 meses os recursos seriam reavaliados.

Uma vez levantado o nível de proficiência dos recursos em cada subsistema, faltava definir o subsistema de cada requisito. Esta informação foi obtida a partir dos programas envolvidos em cada desenvolvimento de requisito, pois existia uma regra de formação para o nome dos programas de cada subsistema.

Com a informação da proficiência do recurso por subsistema e o subsistema de cada requisito, foi possível avaliar se a idéia levantada com o grupo teria estatisticamente alguma correlação com a variação de esforço. Para isso foi traçado o diagrama de dispersão e calculada a correlação entre a variação de esforço (não ajustada) e o nível de proficiência do recurso no subsistema do requisito (figura 5.9).

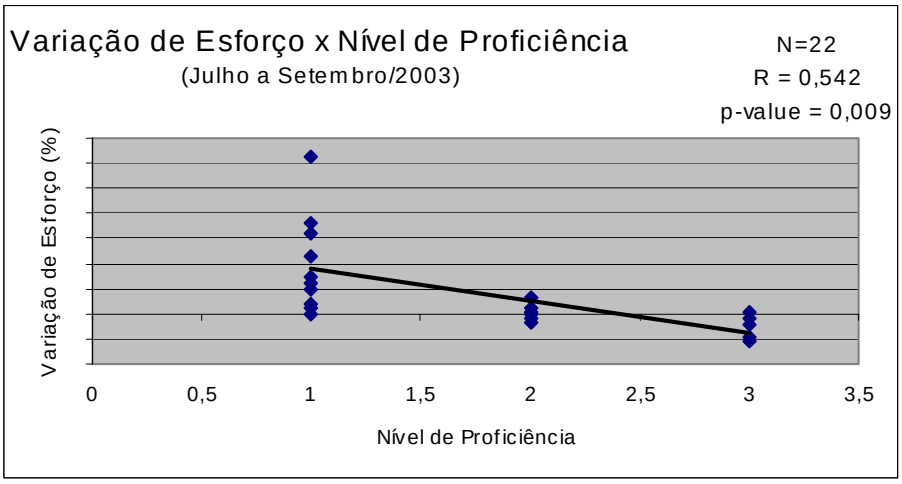

Figura 5-9 - Correlação Variação de Esforço x Proficiência

Para melhor comparar os resultados obtidos pela correlação entre a Variação de Esforço e Nível de proficiência com as correlações encontradas anteriormente, estes dados foram colocados em uma tabela (tabela 5.3).

Variação de Esforço x	Coeficiente de correlação (R)	p-value
Anos de Experiência no Sistema	0,179	0,425
Anos de Experiência em TI	0,326	0,138
Nível de Proficiência	0,542	0,009

Tabela 5-3 - Comparação do resultado das correlações

Comparando os valores do coeficiente de correlação e a significância estatística (*p-value)* destas relações entre variáveis, foi possível constatar que a relação entre a variação de esforço e o nível de proficiência tinha um nível de correlação maior que as demais relações, e que esta correlação era estatisticamente significante (*P-value* menor que 0,05).

Uma vez encontrados indícios de que o nível de proficiência poderia ser um parâmetro melhor que os anos de experiência, foi definido o novo fator de ajuste a ser usado nas estimativas como sendo a equação da linha de tendência do gráfico de relação entre variação de esforço e nível de proficiência, sendo o nível de proficiência o parâmetro da equação.

Para avaliar quais poderiam ser os resultados esperados ao se usar o nível de proficiência do recurso para ajustar as estimativas foi decidido fazer uma simulação, re-estimando-se os requisitos usando o novo fator de ajuste.

Para avaliar se os dados da simulação também teriam um comportamento previsível, foi traçado o gráfico de controle da variação, desta vez utilizando como esforço estimado o

esforço resultante da simulação: a estimativa ajustada pelo novo fator de ajuste – calculado baseado na proficiência:

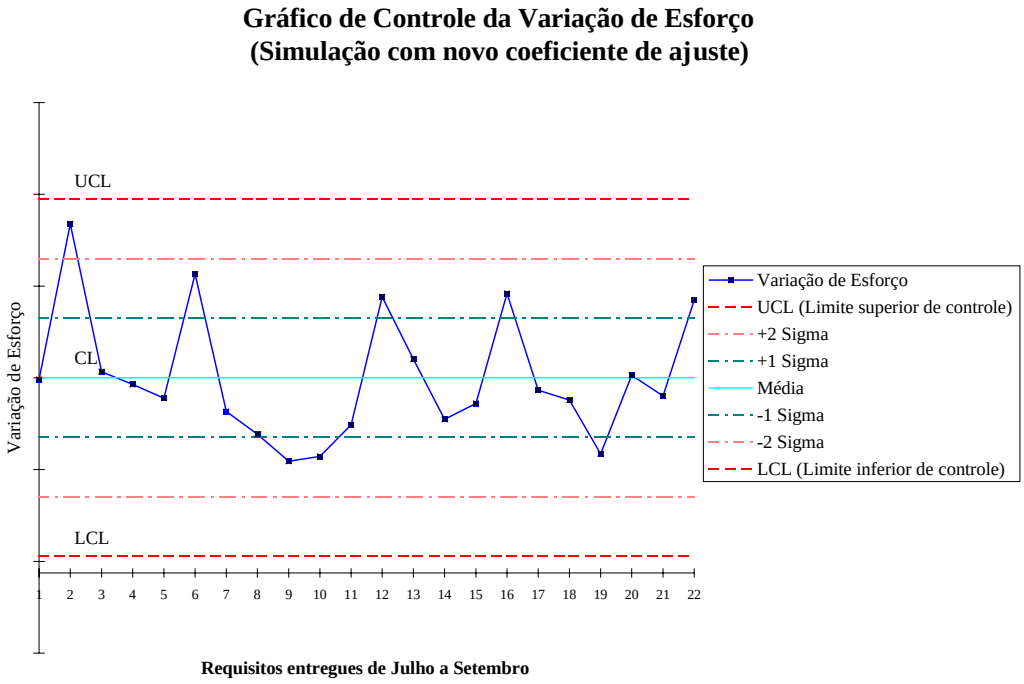

Figura 5-10 - Gráfico de Controle - Simulação

Através do gráfico de controle (figura 5.10) percebeu-se que utilizando-se o novo ajuste o comportamento da variação continuaria estável, pois o processo não seria reprovado em nenhum dos 4 testes que avaliam se o processo está fora de controle .

Concluindo-se que o processo continuaria estável, restava avaliar qual seria o impacto desta mudança na variação de esforço. Para avaliar isto foi calculada a média e a variância em cada uma das amostras (vide tabela 5.4).

	Variação de Esforço Atual	Variação de Esforço Simulada	Diferença
Média	-0,065	0,001	0,064
Variância	0,131	0,119	9%
Amostra	22	22	

Tabela 5-4 - Comparativo variação atual e variação simulada

Comparando os valores obtidos nas duas amostras foram encontrados indícios de que a estimativa utilizando o novo processo de ajuste poderia dar melhores resultados que o processo corrente: a variação média tenderia a zero e a variância diminuiria, aumentando a previsibilidade do processo de estimativa.

Como consequência desta análise, chegou-se à conclusão que o subprocesso que provavelmente mais afetava o processo de estimativa era o sub-processo de ajuste da estimativa. Para melhorar este subprocesso, o novo fator de ajuste começaria a ser utilizado, baseado na proficiência do recurso.

Visando manter o subprocesso de ajuste sob controle, à medida que mais dados fossem acrescentados a equação do ajuste baseado na proficiência seria recalculada, e seria avaliado se a equação se mantém pouco alterada no tempo.

Também continuaria sendo controlado o processo principal, de estimativa, para verificar se com a implementação do novo subprocesso de ajuste o processo de estimativa como um todo continuaria sob controle, e para validar se a melhora esperada se realizaria.

Para complementar a implementação deste processo, os novos atributos identificados foram mapeados no conjunto de medidas do processo:

- Subsistema – Foi mapeado como um elemento do conjunto, definido apenas como uma repartição lógica do sistema de sinistros, tendo como atributo apenas o nome do subsistema e sua descrição.
- Subsistema do Requisito – Este item foi incluído como um atributo de Requisito, relacionado ao elemento subsistema.
- Proficiência por subsistema– A proficiência do recurso em cada subsistema foi mapeada acrescentando-se o tipo "Proficiência" como mais um tipo de nível, e no elemento Nível do Recurso foi acrescentado o atributo subsistema.. Desta forma para cada subsistema foi possível identificar qual o nível de proficiência do recurso (1 – Iniciante, 2- Conhecimento básico, 3- Proficiente ou 4- Especialista).

Com este mapeamento, a proficiência do recurso no subsistema passou a ser usado nas estimativas dos requisitos: dado um requisito do subsistema EDI, por exemplo, se o recurso alocado a este requisito tiver nível 4 de proficiência em EDI a estimativa seria ajustada de acordo com a fórmula.

Para tornar mais claro como ficou a implementação final do conjunto de medidas, a figura 5.11 ilustra o modelo físico de dados implementado no projeto BSCA.

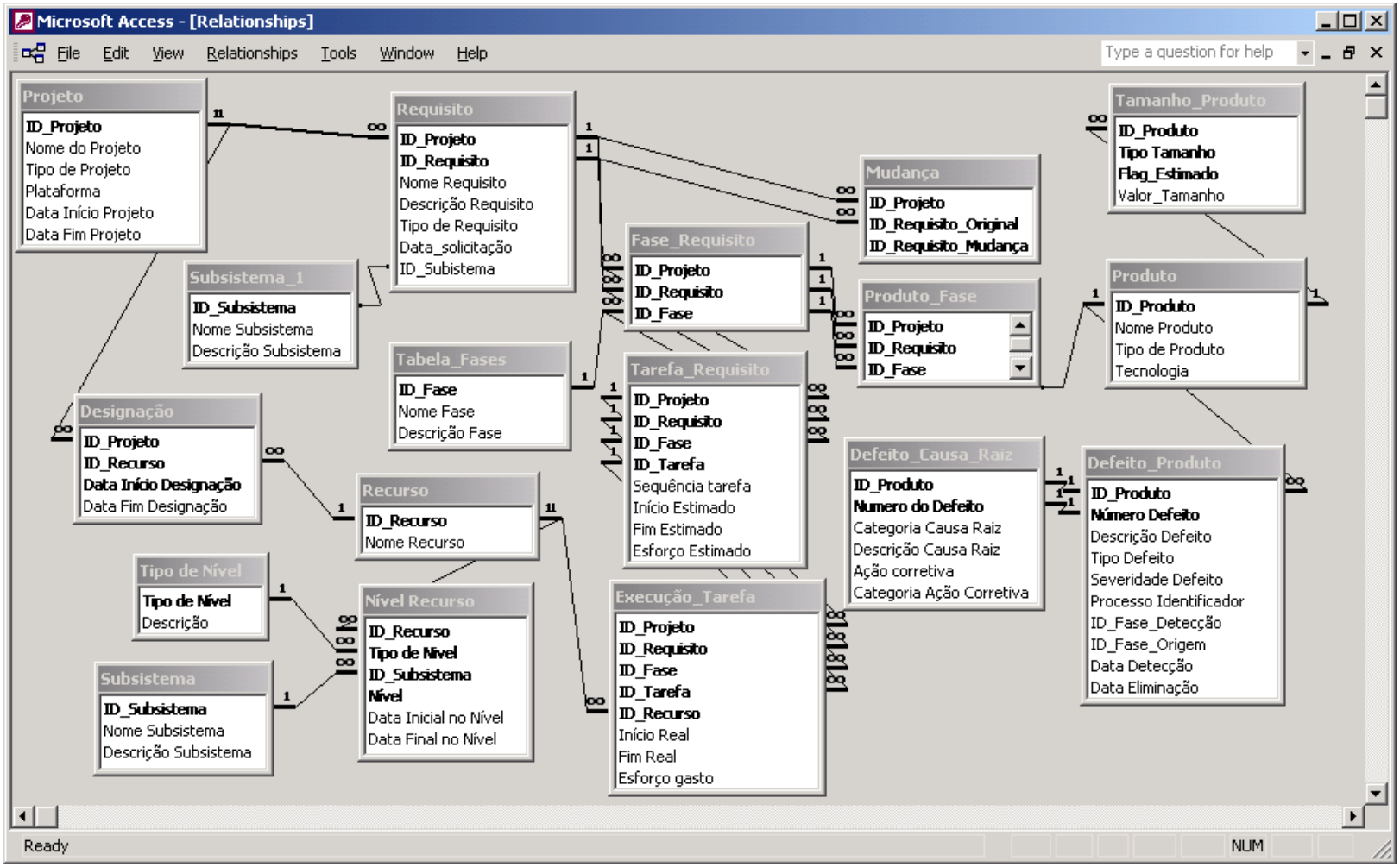

Figura 5-11 - Conjunto de Medidas Final

5.4 AVALIAÇÃO DO ESTUDO DE CASO

A fábrica de Software da EDS Brasil – o *Rio Solution Centre* – foi certificada no nível IV do CMM em Dezembro de 2003, e o projeto BSCA foi um dos projetos que tiveram seu objetivo e sua implementação do controle quantitativo avaliados.

Sem dúvida ter já a grande maioria de seus dados levantados e armazenados em um repositório de dados criado de acordo com o conjunto básico de medidas do processo de desenvolvimento de software facilitou o processo de implementação do controle quantitativo no projeto, pois foi possível rapidamente calcular as correlações entre as medidas envolvidas no processo e seu relacionamento com o objetivo principal do processo. Através da implementação deste conjunto básico de medidas, o projeto BSCA teve dados suficientes para:

- Monitorar os objetivos não priorizados,
- Avaliar o comportamento do processo de estimativa (o processo relacionado ao objetivo definido como prioritário),
- Avaliar a capacidade do processo de estimativa,
- Analisar a correlação entre a variação de estimativa e todas as medidas que eram utilizadas pelos sub-processos da estimativa, num total de 22 correlações.

Outra forma de validar o conjunto de medidas proposto é verificar quantos de seus elementos foram utilizados durante o estudo de caso, demonstrado pela tabela a seguir. Pela tabela 5.5 conclui-se que neste estudo de casos praticamente todos os elementos foram utilizados – uns em maior nível de detalhe, por poderem ter relação direta com o objetivo principal do projeto, outros em menor nível de detalhe, por estarem relacionados a objetivos apenas monitorados. Um ponto a ser considerado nesta avaliação é que objetivos apenas monitorados podem passar a ser controlados dependendo do atingimento do objetivo principal, ou do aumento da criticidade do objetivo.

Elemento	Utilizado?	Comentário
1) Projeto	Não	Não houve necessidade, pois ele só havia sido implementado em um projeto.
2) Recurso	Sim	Monitoramento do objetivo de utilização Relacionamento com o nivel para identificar fatores de ajuste.
3) Requisito	Sim	Nível de detalhe no qual foi controlado o objetivo de redução da variação (mudanças incluídas).
4) Produto	Sim	Usado para relacionar requisito a defeito (objetivo a ser monitorado).
5) Nível	Sim	Utilizado para calculo do fator de ajuste.
6) Fase	Sim	Utilizado para relacionar requisitos a tarefas (somando-se as horas totais estimadas).
7) Tarefa	Sim	Utilizado para relacionar fases à designação (somando-se as horas totais reais, gastas pelos recursos).
8) Defeitos	Sim	Objetivo a ser controlado.
9) Tamanho	Sim	Avaliado como tendo possível correlação com a variação de esforço.
10)Designação	Sim	Utilizado para somar as horas totais gastas no requisito Utilizado para relacionar Recurso ao Requisito, base para se calcular o fator de ajuste da estimativa.

Tabela 5-5 - Elementos do conjunto básico utilizados no estudo de casos

Por outro lado, o estudo de caso mostrou indícios de que não se pode utilizar o conjunto básico de medidas considerando-se que ele terá absolutamente todas as medidas necessárias para se implementar o controle quantitativo. No projeto BSCA, por exemplo, a implementação inicial do conjunto não dispunha dos dados que afinal foram identificados como os que tinham maior correlação com o processo de estimativa neste projeto, a saber, a área do sistema e a proficiência de cada recurso. Neste caso foi importante notar que os novos atributos puderam facilmente ser incorporados ao conjunto e a sua estrutura básica se manteve inalterada, demonstrando a robustez do conjunto.

Um benefício que este projeto não chegou a experimentar foi o tempo que poderia ter sido economizado se este conjunto de medidas tivesse sido implementado desde que o projeto implementou o nível III. Organizações que estão no processo de implementar o nível III do CMM poderiam se beneficiar ainda mais deste conjunto de medidas.

5.5 CONCLUSÃO DO CAPÍTULO

Neste capítulo foi demonstrado como o conjunto básico de medidas do processo de desenvolvimento de software foi mapeado e utilizado em um projeto e como, utilizando-se este conjunto, o projeto implementou o controle quantitativo de processos e obteve a certificação do nível IV do CMM.

O estudo de caso indicou que a implementação deste conjunto de medidas em um projeto é viável, e agrega valor no caminhar para os níveis IV e V do CMM, pois permite que o projeto colete informações sobre o processo de desenvolvimento que poderão revelar-se importantes ao se implementar o controle quantitativo de processos.

Foram levantados indícios de que este conjunto é um conjunto abrangente mas não suficiente de medidas, pois podem haver medidas específicas de alguns projetos que não estão mapeados neste conjunto. No estudo de caso, nem mesmo o próprio projeto tinha consciência destas medidas específicas.

6 CONCLUSÕES

Neste capítulo é apresentado um breve retrospecto deste trabalho, assim como sua conclusão. Além disso, algumas contribuições deste trabalho à ciência da computação são ressaltadas e alguns trabalhos futuros são indicados.

6.1 CONTEXTO DO TRABALHO

O *Capability Maturity Model* (CMM), proposto pela SEI, avalia a maturidade das organizações de software em níveis, do nível I (inicial), onde o sucesso depende de esforços individuais, até o nível V (otimizado), onde a melhoria contínua dos processos é propiciada pela análise quantitativa dos processos.

Este modelo, inicialmente criado para avaliar as empresas prestadoras de serviço para o DOC (Departamento de Defesa dos Estados Unidos), tem sido cada vez mais exigido na indústria da tecnologia da informação como demonstração da maturidade das empresas prestadoras de serviços nesta área.

No Brasil, já existem algumas empresas certificadas nos níveis II e III do CMM. A implementação do nível IV, porém, ainda é um terreno praticamente inexplorado no Brasil.

O CMM preconiza que medidas, inclusive as relacionadas ao processos de software, sejam coletadas no nível III e que no nível IV coloque-se os processos sob controle quantitativo baseado nestas medidas. . Todavia, muitas vezes as medidas coletadas no nível III não são armazenadas no nível de detalhe adequado, nem são armazenadas de maneira apropriada para que seja possível utiliza-las para controlar estatisticamente os processos.

Este trabalho buscou levantar um conjunto básico de medidas do processo de desenvolvimento de software que pudesse ser coletado desde o nível III do CMM, visando facilitar a implementação dos níveis IV e V do CMM, ao prover dados num nível de detalhe suficiente para sua utilização no controle quantitativo de processos .

6.2 COMPARAÇÃO COM OUTROS TRABALHOS DA ÁREA

Em vários trabalhos na área recomenda-se que as medidas sejam definidas em função dos objetivos da organização, em um processo *top-down* (Perkins, 2001). Com isso, seriam levantadas apenas as medidas que satisfizessem a estes objetivos. A idéia de se propor um

conjunto inicial de medidas do processo de software pode sugerir a não aderência a estes princípios, uma vez que não se partiu dos objetivos de uma organização para a proposta do modelo.

Por outro lado, o mesmo Timothy Perkins em seu trabalho "*The Nine Step Metrics Program*" (Perkins, 2001), diz que inicialmente a seleção das medidas pode ser um palpite. Se estes dados atendem necessidades específicas de informações ou não, isto seria determinado pela experiência e que, à medida que o tempo passasse, seria necessária uma revisão da utilidade das medidas inicialmente levantadas. Levando isso em consideração, uma organização poderia levar muito tempo para fazer uma definição inicial das medidas a serem levantadas para o nível IV, para depois poder concluir que o levantamento poderia estar incompleto, ou que as medidas foram armazenadas de forma inadequada, dificultando a análise e o controle estatístico.

Confirmando esta idéia, o *SEI* recomenda que "em adição às necessidades atuais da organização e do projeto, medidas que podem ser úteis no futuro são também coletadas" (Paulk at al, 1993). Desta forma, pode-se concluir que propor um conjunto básico de medidas, que reflita o processo básico de desenvolvimento de software de qualquer organização pode ser opção válida à abordagem *top-down*.

6.3 PRINCIPAIS CONTRIBUIÇÕES

Neste trabalho foi apresentada uma visão geral do CMM, ressaltando-se requisitos específicos para o nível IV do CMM, especialmente o controle quantitativo dos processos.

O presente trabalho também fez uma introdução ao controle estatístico de processos (SPC – *Statistical Process Control*), definindo termos básicos e interpretação das principais ferramentas utilizadas, bem como alguns aspectos e problemas que foram enfrentados por uma organização ao iniciar a implantação do controle estatístico de processos..

O objetivo principal do trabalho era a definição de um conjunto básico de medidas do processo de desenvolvimento de software, que estivesse num nível de detalhe suficiente para auxiliar o processo de implementação do controle quantitativo dos processos em uma organização. A premissa básica era que seria possível levantar estas medidas no próprio CMM, analisando-se as KPAs do CMM relacionadas à engenharia de software.

Levantado este conjunto básico de medidas do processo de desenvolvimento de software, o mesmo foi utilizado por um projeto que implementou o controle quantitativo de processos, e participou da certificação da fábrica de software da EDS Brasil como nível IV do CMM. Esta utilização serviu como estudo de caso da validade, viabilidade e abrangência deste conjunto de medidas.

Através do estudo de caso, foi mostrado que é viável o levantamento deste conjunto de medidas proposto, e que a implementação deste conjunto de medidas pode auxiliar o gerenciamento quantitativo e melhoria contínua dos processos de engenharia de software.

A utilização do conjunto de medidas em um projeto deu indicativos que este conjunto seria abrangente, pois quase todas as medidas necessárias à implementação do controle quantitativo no projeto faziam parte: do conjunto inicialmente proposto. Deste conjunto inicial algumas medidas foram apenas monitoradas, por não terem relação direta com o objetivo principal do projeto, enquanto que outra parte considerável das medidas foi utilizada para colocar sob controle o objetivo principal do projeto e para identificar uma possível melhoria no processo. Neste levantamento o nível de detalhe das medidas pôde ser considerado adequado, pois as relações entre as variáveis puderam facilmente ser identificadas.

Outro benefício deste trabalho foi levantar que a proposta de um conjunto básico de medidas pode ser uma alternativa válida à abordagem *top-down,* que sugere que só se coletem as medidas relacionadas aos objetivos da organização. O benefício desta proposta em relação à abordagem *top-down* é que se as medidas já forem coletadas em suficiente nível de detalhe desde a implementação na organização do CMM nível III, ao se começar a caminhar para o nível IV a organização já teria os dados para conseguir avaliar o desempenho dos seus processos para, de acordo com os objetivos definidos, colocá-los sob controle e, mais tarde, identificar as causas comuns de variação nestes processos para a implementação de melhorias, base para se atingir o nível V do CMM.

Por outro lado, foi possível entender que a implementação do gerenciamento quantitativo dos processos pressupõe uma quebra dos processos principais em sub-processos. Esta decomposição pode identificar medidas muito específicas de um projeto, que talvez não constem do conjunto de medidas aqui proposto.

Sintetizando as conclusões, neste trabalho foi possível perceber indícios de que a proposta de um conjunto básico de medidas para a implementação do controle quantitativo de processos é uma alternativa válida à abordagem tradicional (*top-down*), que é viável a implementação deste conjunto, que este conjunto de medidas levantado é abrangente e possui um nível de detalhe suficiente, auxiliando na implementação do controle quantitativo de processos e que ele pode ser considerado um conjunto de medidas básicas a serem implementadas em uma organização, sabendo-se que pode ser necessária a inclusão de medidas específicas de algum subprocesso muito específico de um projeto.

6.4 TRABALHOS FUTUROS

Durante o processo de implementação do controle quantitativo nos projetos, foi notado que outros projetos também tinham subprocessos importantes que se relacionavam ao nível de conhecimento do recurso em uma área específica. Outro projeto também classificou este conhecimento em quatro níveis semelhantes aos utilizados no projeto BSCA. É possível que o ajuste feito para o projeto BSCA seja um acréscimo válido ao conjunto de medidas, pois da mesma forma que esta informação foi relevante em três projetos do *RioSC*, ela pode ser relevante em outros projetos.

Uma vez atualizado o conjunto de medidas com estes novos campos, ele será implantando em outros projetos do *RioSC*. Desta forma será possível calcular o percentual de medidas relevantes que já se encontravam no conjunto, bem como criar um *ranking* de medidas menos e mais utilizadas. Como fruto desta análise será possível analisar a abrangência do conjunto quantitativamente, bem como analisar se existem medidas que freqüentemente obtêm alta correlação com os processos de desenvolvimento de software, nas quais todos os projetos deveriam se focar.

Outro trabalho futuro é a adaptação deste conjunto ao novo modelo da SEI, o CMMI. Como neste novo modelo a SEI detalha bastante o processo de desenvolvimento, quebrando a KPA de Engenharia de Produto em cinco áreas de processo, pode ser necessário alguma alteração. Mais uma vez, através desta adaptação será possível avaliar novamente a robustez do conjunto.

O conjunto de medidas foi sumarizado em um modelo relacional. Este modelo é uma contribuição para a área de engenharia de software, diminuindo etapas de modelagem em

organizações que utilizem este tipo de ferramenta. Todavia, um outro trabalho futuro poderia sugerir um tipo de implementação mais adequado: a abordagem dimensional. Utilizando-se esta abordagem haveria maior flexibilidade para alterar tipos de medidas e incluir perspectivas, e provavelmente novas relações poderiam ser encontradas.

Focalizando a área de Controle Estatístico do Processo (SPC): uma vez que o conjunto tem definidas as medidas e seus domínios, outro possível trabalho é a definição de ferramentas apropriadas para controlar cada uma das variáveis do processo de desenvolvimento de software. Em uma organização nem todos as medidas serão colocadas sob controle estatístico. Todavia, conforme descrito por Gupta (2002, p.8), às vezes gasta-se um tempo considerável definindo-se que gráficos de controle seriam apropriados para cada medida. Ter essa definição de ferramentas pré selecionadas poderia facilitar a implementação do controle estatístico do processo em organizações de software.

Indo mais além na área de SPC, este conjunto de medidas poderia ser utilizado com base para um *framework* para a implantação do controle quantitativo de processos, utilizando-se uma abordagem inspirada no *Tame project* (Basili, Oivo, 1992, p.886-898).

Assumindo que este conjunto de medidas é abrangente, e inclui medidas dos principais processos de desenvolvimento de software, pode-se também assumir que provavelmente os objetivos que um projeto de desenvolvimento de software possa levantar teriam como meta o controle de uma das medidas deste conjunto. Então a primeira função do *framework* seria permitir a definição de objetivos e a seleção da medida do conjunto básico relacionada ao objetivo que se quer controlar. Entendendo que este conjunto já estará alimentado com dados do projeto, o *framework* já poderia mostrar se o processo relacionado ao objetivo está ou não sob controle. Se o processo estiver sob controle, o framework solicitaria que se entrassem com os limites requeridos no objetivo (USL e LSL). Com esta informação seria possível avaliar se o processo seria ou não capaz.

Se o processo não fosse capaz, o *framework* passaria a calcular possíveis relações entre as variáveis do conjunto e a medida a ser controlada. Isso seria possível porque todas as possíveis relações entre elementos deste modelo, e consequentemente entre seus atributos, já estariam definidas a *priori*, em uma tabela de relações.. Uma vez calculadas as possíveis relações, o framework poderia indicar qual relação demonstraria maior correlação e significância estatística. Logicamente o aceitar desta correlação dependeria da análise do usuário. No estudo de caso, por exemplo, se a correlação entre a variação de esforço e os

anos de experiência tivesse se mostrado significativa, teria que se acreditar que a variação de esforço aumenta à medida que o recurso se torna mais experiente.

Como o próprio conjunto básico de medidas proposto não assume ter todas as medidas possíveis, o *framework* também teria que ser customizável. Para isso, além da customização do conjunto básico de medidas, as possíveis relações entre as novas medidas teriam que ser acrescentadas na tabela de relações.

7 REFERÊNCIAS BIBLIOGRÁFICAS

Ares, J., Pazos, J. – **Conceptual modelling: an essential pillar for quality software development** – Knowledge Based Systems 11, 1998.

Basili, V., Oivo, M. – **Representing Software Engineering Models: the TAME Goal Oriented Approach** – IEEE Transactions on Software Engineering, Vol. 18, No. 10, Outubro 1992.

Florac, William A. – **Software Quality Measurement: A Framework for Counting Problems and Defects** – (CMU/SEI-92-TR-022 - ESC-TR-92-022), Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa, 1992.
Disponível em www.sei.cmu.edu/pub/documents/ 92.reports/pdf
Último acesso em Fev/2004.

Florac, William A, Park, Robert E., Carlton, Anita D. – **Practical Software Measurement: Measuring for Process Management and Improvement** – (CMU/SEI-97-HB-003), Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1997.
Disponível em www.sei.cmu.edu/pub/documents/ 97.reports/pdf
Último acesso em Fev/2004.

Florence, Al. –**Lessons Learned in Attempting to Achieve Software CMM Level 4 –** CrossTalk – The Journal of Defense Software Engineering, Agosto 2001.

Gupta, Rajesh – **Use of Statistical Process Control in the Software Process – An Experiential Learning at Satyam** – SEPG 2002.
Disponível em www.qaiindia.com/SEPG_minisite/ sepg2003/mailers
Último acesso em Fev/2004.

IEEE Software Engineerign Standards - **IEEE Standard Glossary of Software Engineering Terminology** - Standard 610.12-1990, Std 610.12-1990 Fevereiro 1991.
Disponível em standards.ieee.org/catalog/olis/se.html.
Último acesso em Fev/2004

Jones, Capers – **Software Measurement Programs and Industry Leadership** - CrossTalk – Fevereiro 2001.

Leavengood, S., Reeb, J. – **How and Why SPC Works** – in Performance Excellence in the Wood Products Industry – Junho 1999.
Disponível em http:\\wood.oregonstate.edu.
Último acesso em Fev/2004.

Mellor, S., Shlaer, S. – **Object Oriented System Analysis** – Modeling the World in Data – Prentice Hall – 1988

Ministéria da Ciência e Tecnologia - **Qualificação CMM no Brasil**
Disponível em http://www.mct.gov.br/Temas/info/Dsi/qualidad/CMM.htm.
Último acesso em Fev/2004.

Montgomery, D. C. – **Introduction to Statistical Quality Control**, 3ª edição – John Wiley & Sons, Nova York, 1997.

Pall, Gabriel A.- **Quality Process Management** – Englewood Cliffs, NJ, Prentice Hall, 1987 *Apud* Florac, William A, Park, Robert E., Carlton, Anita D. **Practical_Software_Measurement: Measuring for Process Management and Improvement** – (CMU/SEI-97-HB-003), Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa, 1997.
Disponível em www.sei.cmu.edu/pub/documents/ 97.reports/pdf/
Último acesso em Fev/2004.

Paulk, Mark C., Curtis, Bill et al. – **Capability Maturity Model for Software, Version 1.1** – Software Engineering Institute, CMU/SEI-93-TR-24, Fevereiro 1993.
Disponível em http://www.sei.cmu.edu
Último acesso em Jan/2004.

Paulk, Mark C., Weber, Charles V. et al. – **Key Practices of the Capability Maturity Model, Version 1.1** – Software Engineering Institute, CMU/SEI-93-TR-25, Fevereiro 1993.
Disponível em http://www.sei.cmu.edu
Último acesso em Jan/2004.

Paulk, Mark C. – **Toward Quantitative Process Management With Exploratory Data Analysis** – International Conference on software Quality – 1999.
Disponível em http://www.sei.cmu.edu/pub/cmm/high-maturity
Último acesso em Mar/2004.

Perdue, Jeff. – **Why is level 4 So Hard?** – Washington D.C., Software Process Improvement Network, Novembro 2000.

Perkins, Timothy K. – **The Nine Step Metrics Program** – CrossTalk – The Journal of Defense Sofware Engineering, Fevereiro 2001.

Pressman, Roger S. – **Software Engineering: A Practitioner's Approach** - McGraw-Hill, 2000.

Putnam, Lawrence H. – **Measures for Excellence** – Yourdon Press Computing Series, 1994.

Ramos, Alberto Wunderley – **Controle Estatístico de Processo**. Em Contador, José Celso et al, **Gestão de Operações: A Engenharia de Produção a serviço da Modernização da Empresa.** São Paulo, Ed Edgar Blucher, 1997.

Rosemberg, Linda – **Software Process Assesment (SPA)** – NASA Software Assurance Technology Center.
Disponível em http://satc.gsfc.nasa.gov/support/SMO_NOV94/spa.PDF
Último acesso em Fevereiro/2004.

Shewhart, Walter S – **Economic Control of Quality of Manufactured Product** - Nova York – NY – Van Nostrand –1931 (re-impresso por American Society of Quality Control – 1980).

Stutzk, Richar D. – Measuring Integrated Product teams – ESCOM 2001 - London, England , Abril 2001.
Disponível em http:// sunset.usc.edu/events/2002/cocomo17/Measuring
Último acesso em Fev/2004.

OMG. – **Unified Modeling Language Specification**
Disponível em http://www.omg.org
Último acesso em Nov/2003

APÊNDICES

APÊNDICE 1 - DEFINIÇÃO DO MODELO DE DADOS BASEADO NO CONJUNTO BÁSICO DE MEDIDAS DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

A título de consulta, este Apêndice detalhará como foi feita a modelagem dos elementos do conjunto de medidas do processo de desenvolvimento de software em um modelo de dados relacional.

As primeiras entidades foram facilmente identificáveis, pois foram extraídas diretamente dos itens identificados como parte do conjunto de medidas: Projeto, Recurso, Nível, Requisito, Fase, Tarefa, Defeito, Mudança e Produto .

Ao modelar a entidade Nível, foi possível concluir que este deveria ser modelado como entidade fraca de Recurso, já que este é praticamente um atributo de recurso. Continua sendo uma entidade separada porque o recurso pode ser classificado em mais de um tipo de nível, e pode passar por mais de um nível dentro de um projeto – ao ser promovido, por exemplo, e para fins de análise do histórico é importante saber o nível da pessoa quando trabalhou em um requisito.

Continuando no processo de estruturação do modelo, pôde-se perceber que Defeito poderia ser considerado atributo de Produto, mas como um produto pode ter mais que um defeito, Defeito também deveria ser modelado como entidade fraca. Também se encaixam neste caso a entidade Fases, dependente de Requisito, e a entidade Tarefa, dependente de Fase.

Também foi possível identificar que Mudança poderia ser considerado um autorelacionamento com requisito, pois todos os atributos são comuns, exceto a que requisito a mudança se refere.

Não era tão claramente notado, mas também foi identificado que Tamanho tinha seus atributos próprios, como se era estimado ou real, o tipo de tamanho (linhas de código, pontos de função, etc...). Também, para fins de análise, é possível medir-se mais de um tamanho para cada produto, como medir-se pontos de função e linhas de código para um software. Por este motivo Tamanho também foi modelado como entidade fraca de Produto.

Sumarizando-se a estruturação em entidades, as seguintes entidades próprias foram definidas:

1) Projeto
2) Recurso,
3) Requisito
4) Produto.

Também foram definidas as seguintes entidades fracas no modelo:

5) Nível (dependente de Recurso),
6) Mudanças (dependente de Requisito),
7) Fase (dependente de Requisito),
8) Tarefa (dependente de Fase),
9) Defeitos (dependente de Produto),
10) Tamanho (dependente de Produto)

Definidas as entidades, passou-se à definição de relacionamentos.

De acordo com a metodologia adotada na organização, o **Projeto** tem um Grupo de **recursos designados** a si, com a finalidade de implementar os **Requisitos** levantados. Podem ser solicitados **Requisitos** que promovam mudanças em **Requisitos** em andamento. O desenvolvimento dos Requisitos *se estrutura* em **Fases,** que *constróem* diversos **Produtos** derivados do processo de engenharia de software. Cada **Produto** *é medido em* **Tamanhos,** estimado e real, seja este produto um documento, cujo tamanho seria medido em páginas de documentação; ou software, medido em pontos de função e/ou linhas de código, por exemplo. O **Produto** gerado *pode conter* **Defeitos**, que terão uma série de atributos definidos visando identificá-los e classificá-los, de forma que a organização possa identificar razões em comum para os defeitos e propor melhorias no processo. As **Fases** se subdividem em **Tarefas.** Cada **Tarefa** tem um esforço e duração estimados e, baseado nisto, os **Recursos** disponíveis são colocados à disposição para executarem as tarefas.

Descrevendo-se o relacionamento entre **Recurso** e as **Tarefas** de uma fase, foi possível perceber a necessidade de obter mais informações sobre este relacionamento, como a data em que

o recurso começou e terminou o trabalho na tarefa e o esforço gasto pelo recurso. Sendo assim, foi também modelada a entidade **Execução**, resultante do relacionamento entre tarefa e recurso.[6]

EXECUÇÃO		
Conceito	Associação de recursos às tarefas do projeto.	
Atributo	**Definição**	**Valor**
Data início	Data de início da associação do recurso na tarefa.	Data.
Data fim	Data final da associação do recurso na tarefa.	Data.
Esforço real	Esforço real gasto pelo recurso para completar a tarefa.	Numérico.

Tabela Apêndice 1-1 - Conceito/Atributos/Valores de Execução

Com este novo elemento modelado, percebeu-se que uma vez estando o esforço real modelado na entidade Execução, o esforço real total gasto em uma tarefa poderia ser calculado a partir da soma do esforço gasto pelos recursos assinalados na tarefa. O mesmo se aplicaria à entidade Fase: a soma do esforço gasto em todas as tarefas da fase já calcularia o esforço real da Fase. Portanto, não era mais necessário modelar esta informação na entidade Tarefa nem Fase. Pelo mesmo raciocínio, não era necessário incluir o atributo duração real nestas entidades.

[6] Através deste modelo, é possível identificar que recursos trabalharam em que requisito. O objetivo desta informação é obter dados sobre a curva de aprendizado, impacto do nível do recurso na produtividade, etc. Todavia, o acesso a esta informação deve estar protegido, para se garantir que as medidas não serão utilizadas para avaliar o desempenho de um indivíduo, como colocado no compromisso 1.2 da KPA de *Quantitative Process Management* (Paulk et al, 1993).

APÊNDICE 2 - CTQ TREE

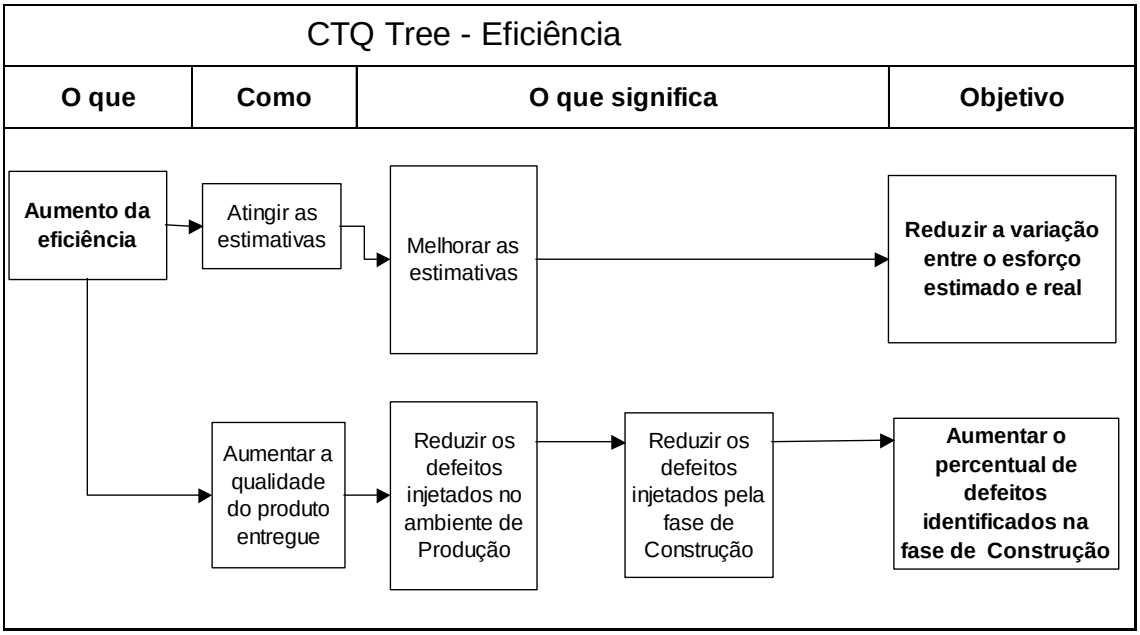

Figura Apêndice 2-1 - CTQ Tree - Eficiência

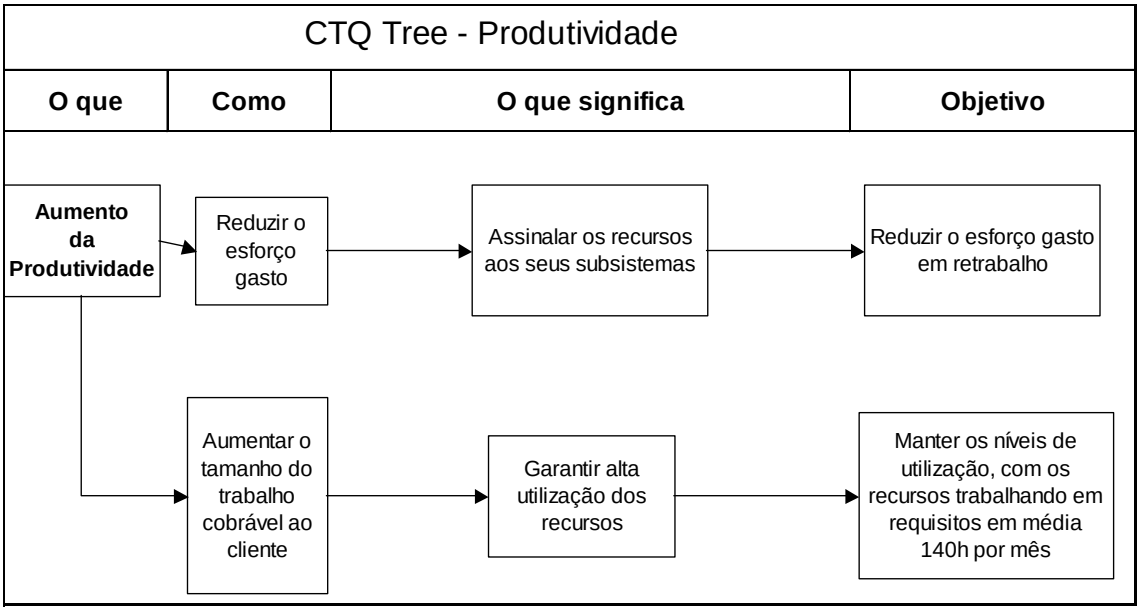

Figura Apêndice 2-2 - CTQ Tree - Produtividade