UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

ESCOLA POLITÉCNICA

DCC/SEGRAC

GERENCIAMENTO DE DESEMPENHO DE PROJETOS DE CONSTRUÇÃO DE SOFTWARE: APLICAÇÃO DE MÉTRICAS, MEDIÇÃO E BOAS PRÁTICAS

Halan Ridolphi Alves

2008

GERENCIAMENTO DE DESEMPENHO DE PROJETOS DE CONSTRUÇÃO DE SOFTWARE: APLICAÇÃO DE MÉTRICAS, MEDIÇÃO E BOAS PRÁTICAS

Halan Ridolphi Alves

Monografia apresentada no Curso de Pós-Graduação em Gerenciamento de Projetos, da Escola Politécnica, da Universidade Federal do Rio de Janeiro.

Orientador:

Heraldo Luís Silveira de Almeida

Rio de Janeiro

Março, 2008

GERENCIAMENTO DE DESEMPENHO DE PROJETOS DE CONSTRUÇÃO DE SOFTWARE: APLICAÇÃO DE MÉTRICAS, MEDIÇÃO E BOAS PRÁTICAS

Halan Ridolphi Alves

Orientador:
Heraldo Luís Silveira de Almeida

Monografia submetida ao curso de Pós-graduação em Gerenciamento de Projetos, da Escola Politécnica, da Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção do título de Especialista em Gerenciamento de Projetos.

Aprovado por:

_____
Prof. Eduardo Linhares Qualharini – Presidente

_____
Prof. Lysio Séllos da Costa Filho

_____
Prof. Heraldo Luís Silveira de Almeida

Rio de Janeiro
Março, 2008

ALVES, Halan Ridolphi.
Gerenciamento de Desempenho de Projetos de Construção de Software: Aplicação de Métricas, Medição e Boas Práticas / ALVES, H. R. Rio de Janeiro: UFRJ/POLI, 2008.
viii, 60f. il.; 29,7cm.
Orientador: Heraldo Luís Silveira de Almeida.
Monografia (especialização) – UFRJ, Escola Politécnica, Curso de Pós-Graduação em Gerenciamento de Projetos, SEGRAC, 2008.
Referências Bibliográficas: f. 50
1. Gerenciamento de Projetos de Software. 2. Medição de Software 3. Métricas de Software. – Monografia.
I. ALMEIDA, H. L. S. de.
II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Pós-Graduação em Gerenciamento de Projetos.
III. Título.

Destacamos nossos sinceros agradecimentos à equipe de professores do curso de pós-graduação em Gerenciamento de Projetos ministrado pela Escola Politécnica da UFRJ, pelas inestimáveis e duradouras lições que nos transmitiram. Nestes quase dois anos de parceria e convivência constatamos que a troca de informações e experiências com a equipe de professores da POLI, pôde proporcionar-nos muito maior sensatez, acurácia, percepção, destreza e habilidades profissionais. Agradecemos a todos que cooperaram e interagiram conosco durante este período possibilitando nosso aperfeiçoamento profissional e pessoal. Ademais, acreditamos que também tenhamos transmitido alguma boa mensagem a cada um dos professores durante esta nossa convivência. Expressamos nosso reconhecimento ao professor doutor Heraldo Almeida, pela orientação segura, paciente, crítica e solidária, estimulando-nos com a confiança de estar no rumo correto, incentivando-nos nos momentos de dificuldades e dúvidas, enfim, contribuindo efetivamente para que este estudo ganhasse um nível maior de qualidade. Ressaltamos nossa gratidão ao pessoal da secretaria do curso, o qual sempre nos atendeu com consideração, profissionalismo e solidariedade. Portanto, desde já, temos somente boas lembranças deste período de cooperação, estudo e intercâmbio de conhecimentos.

RESUMO

GERENCIAMENTO DE DESEMPENHO DE PROJETOS DE CONSTRUÇÃO DE SOFTWARE: APLICAÇÃO DE MÉTRICAS, MEDIÇÃO E BOAS PRÁTICAS

Halan Ridolphi Alves

Orientador:
Heraldo Luís Silveira de Almeida

Resumo da Monografia submetida ao corpo docente do curso de Pós-graduação em Gerenciamento de Projetos – Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção do título de Especialista em Gerenciamento de Projetos.

O objetivo do trabalho é focar na utilização de boas práticas e técnicas viáveis de estimativas de tamanho, métricas de desempenho e processo de medição em projetos de desenvolvimento de software e, neste sentido, demonstrar os benefícios da realização de processos operacionais de apoio ao controle de projetos de software.
A estratégia de elaboração do trabalho é explorar e combinar conhecidas técnicas e métodos em uso na indústria de software, tais como, PMBOK, SWEBOK, CMMI, MPS.BR, ISO 12207, ISO 15939, AMI, COCOMO II, APF, GQM, PDCA, PCU, BSC, reunindo as boas práticas e sugerindo a adoção de padrões e procedimentos sistemáticos e disciplinados de suporte a medição e avaliação de progresso de projetos de construção de software.

Palavras-chave: Gerenciamento de Projetos de Software, Medição de Software, Métricas de Software.

Rio de Janeiro
Março, 2008

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE GRÁFICOS

LISTA DE FÓRMULAS

LISTA DE SIGLAS, ABREVIATURAS E REDUÇÒES

Sigla	Descrição
5S	Programa de Qualidade: utilização, ordenação, limpeza, saúde ou autodisciplina
6σ	Seis Sigma
ABNT	Associação Brasileira de Normas Técnicas
APF	Análise de Pontos de Função
AMI	*Assess, Analyse, Metricate, Improve (The ami Handbook)*
BSC	*Balanced Scorecard*
CASE	*Computer Aided Software Engineering*
CMMI	*Capability Maturity Model Integrated*
COCOMO	*Constructive Cost Model*
EVA	*Earned Value Analysis*
FCS	Fator(es) Crítico(s) de Sucesso
GQM	*Goal-Question-Metric*
IEEE	*Institute of Electrical and Electronic Engineers*
IFPUG	*International Function Point Users Group*
ISO	*International Organization for Standardization*
ISO 12207	Norma ISO para processos de ciclo de vida de software
ISO 15939	Norma ISO para processos de medição de software
KLOC	*Kilo Lines of Code (one thousand source lines of code)*
MPS.BR	Melhoria de Processos do Software Brasileiro
NBR	Norma Brasileira
PCU	Pontos de Caso de Uso
PDCA	*Plan-Do-Check-Act*
PMBOK	*Project Management Body of Knowledge*
PMI	*Project Management Institute*
POLI	Escola Politécnica

Sigla	Descrição
ROI	*Return Of Investiment*
PSM	*Practical Software Measurement*
RUP	*Rational Unified Process*
SEI	*Software Engineering Institute*
SEGRAC	Núcleo de Pesquisa em Ciências da Engenharia
SLOC	*Source Lines Of Code*
SWEBOK	*Software Engineering Body of Knowledge*
UFRJ	Universidade Federal do Rio de Janeiro
TI	Tecnologia da Informação

GLOSSÁRIO

Termo	Definição
Medição de Software	É um processo que visa fornecer aos gerentes um conjunto de dados úteis e tangíveis para dimensionar, estimar, planejar e controlar os projetos com rigor e precisão.
Gerenciamento de Projetos	É a aplicação de conhecimentos, habilidades, ferramentas e técnicas na execução de atividades relacionadas ao projeto, com intuito de atingir seu conjunto de objetivos pré-definidos.
Engenharia de Software	É uma disciplina da engenharia que se ocupa de todos os aspectos da produção de software.
Processo de Software	É um conjunto de atividades cuja meta é o desenvolvimento ou a evolução de software.
Análise de Pontos de Função	Método padrão para medir software do ponto de vista do usuário pela quantificação da funcionalidade fornecida.

1 INTRODUÇÃO

1.1 Motivação da Pesquisa

A indústria de construção de software recorre a uma ampla variedade de técnicas, métodos, ferramentas, normas e processos da engenharia de software visando responder os seguintes desafios em gerenciamento de projetos de software:

- Por que o prazo estimado para construção de um produto de software geralmente é superado?
- Por que o grau de progresso na construção de um sistema de software é difícil de medir?

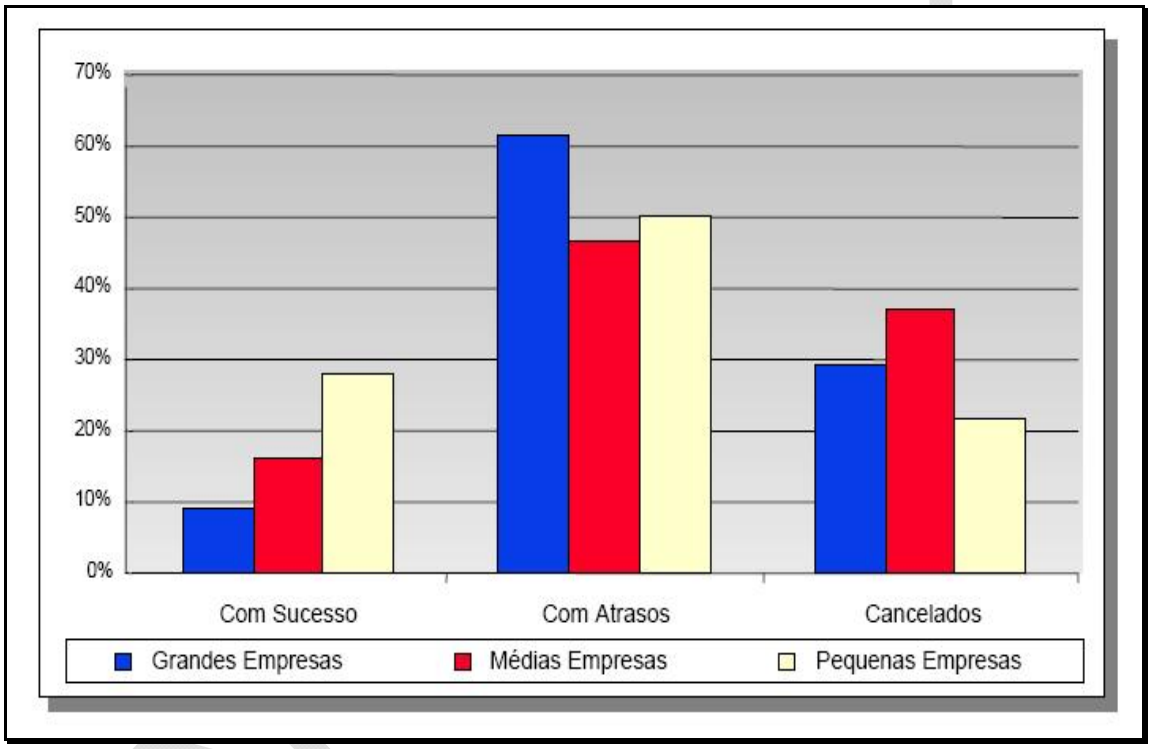


Gráfico 1: Estatísticas de Capacidade de Conclusão de Projetos de Software

Fonte: Chaos Reports, Standish Group Inc., 1995

Dentre as metas fundamentais da Engenharia de Software destaca-se a necessidade de transformação da produção de sistemas de software de uma maneira artística, indisciplinada, informal e pouco entendível para uma abordagem devidamente controlada, sistemática, quantificada e previsível – incluindo o gerenciamento de entidades técnicas e sociais – visto que, com freqüência, essa é maneira mais eficaz de produzir um produto agregando alta qualidade, afirma Sommerville (2007). Há mais de 20 anos a comunidade de engenharia de software tem discutido e pesquisado acerca de medição de software e, no entanto, comumente, não é verificada sua utilização, na prática, pela grande maioria dos projetos de construção de software. Pesquisas realizadas em empresas de software indicam que mais da metade de grandes projetos de software se deparam com algum tipo de

atraso, excesso de custo ou prazo ou algum fracasso na operação do produto final após implantado no ambiente operacional alvo do cliente.

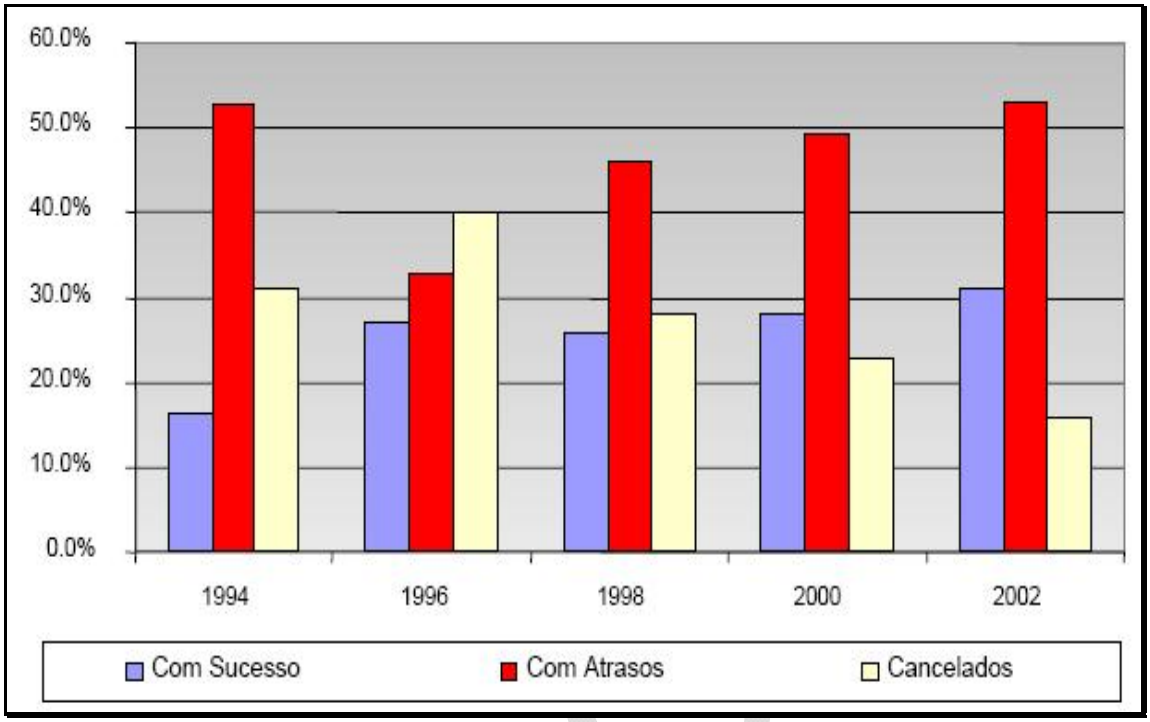


Gráfico 2: Estatísticas de Conclusão de Projetos de Software

Fonte: Chaos Reports, Standish Group Inc., 2004

Pfleeger (2003) reconhece que a produção de software está continuamente experimentando diversas reformulações metodológicas e tecnológicas, ao mesmo tempo em que precisa superar antigos problemas, como baixa qualidade, altos custos de desenvolvimento, ausência de uma notação técnica universal, demora na entrega do produto e baixa satisfação dos clientes. Ademais, as organizações de software já estão observando seriamente a importância de se reformular periodicamente as estratégias para competir, sobreviver e sustentar a posição no mercado, de forma que os clientes percebam diferenciais nos produtos e serviços em termos de preço, entrega, desempenho, valor agregado e qualidade. Isto reforça a observação de Peter Drucker, de que "*não existe gerenciamento sem medidas*", ou seja, que a medição da qualidade e o exercício de melhorias contínuas nos processos produtivos vêm se destacando entre as principais metas e ativos organizacionais. Para Pressman (2002), as organizações necessitam exercitar-se na disciplina de controle para obter gradativamente padrões superiores de maturidade, alavacando sua competividade e capacidade de produção de bens e serviços que atendam ou mesmo superam as expectativas dos clientes. Ademais, aquisição de maturidade é uma meta volátil, visto que seus FCS descritos em termos de tecnologia, metodologia e gestão mudam

continuamente em resposta às novas necessidades e exigências impostas pelos mercados, negócios e indivíduos.

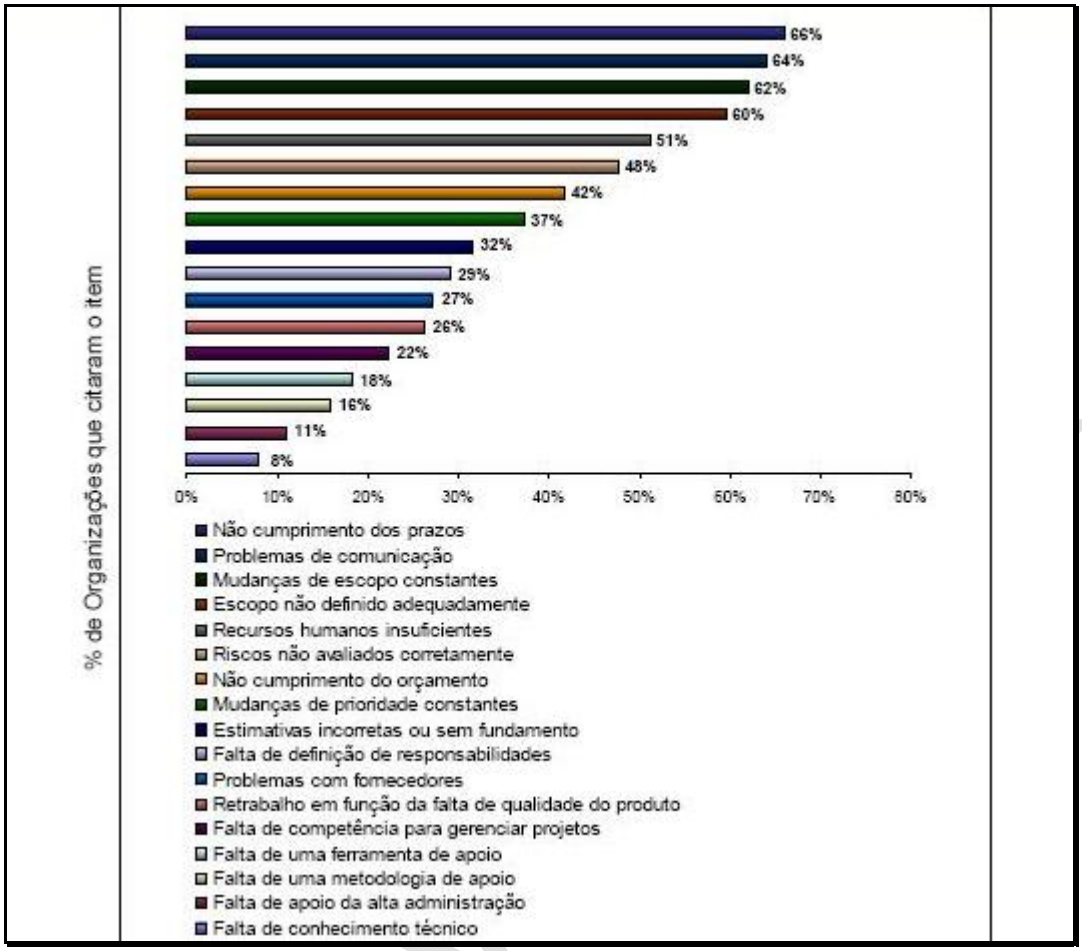


Gráfico 3: Deficiências nas Empresas em Gerenciamento de Projetos

Fonte: Estudo de Benchmarking em Gerenciamento de Projetos Brasil, PMI – Chapters Brasileiros, 2007

No sentido de enfatizar os diferenciais competitivos das organizações, diversas metodologias de gestão tática e estratégica, de maturidade, capacidade e otimização produtiva têm-se popularizado, prescrevendo e elecando uma série de regras para melhoria da eficiência operacional e eficácia dos negócios, cada qual enfatizando aspectos relevantes da organização, quer em relação aos seus processos produtivos e gerenciais internos (incluindo infra-estrutura tecnológica), quer em relação ao seu capital humano (preferencialmente, qualificado, disponível e com qualidade de vida), quer em relação ao seu relacionamento com o mercado consumidor (fidelização, satisfação, market share) e a cadeia de suprimentos.

Neste contexto, o estudo pretende discorrer acerca da utilização de boas práticas e técnicas viáveis de estimativas de tamanho, métricas de desempenho e processo de medição em projetos de construção de software e, propor aos gerentes de projetos a execução de processos racionais e controláveis que auxiliem efetivamente no controle de projetos de software. A proposta não é esgotar a mensuração de desempenho em

gerenciamento de projetos, mas tão somente cooperar na divulgação de conhecimentos e métodos que estão sendo utilizados na prática corporativa, ressaltando os benefícios obtidos.

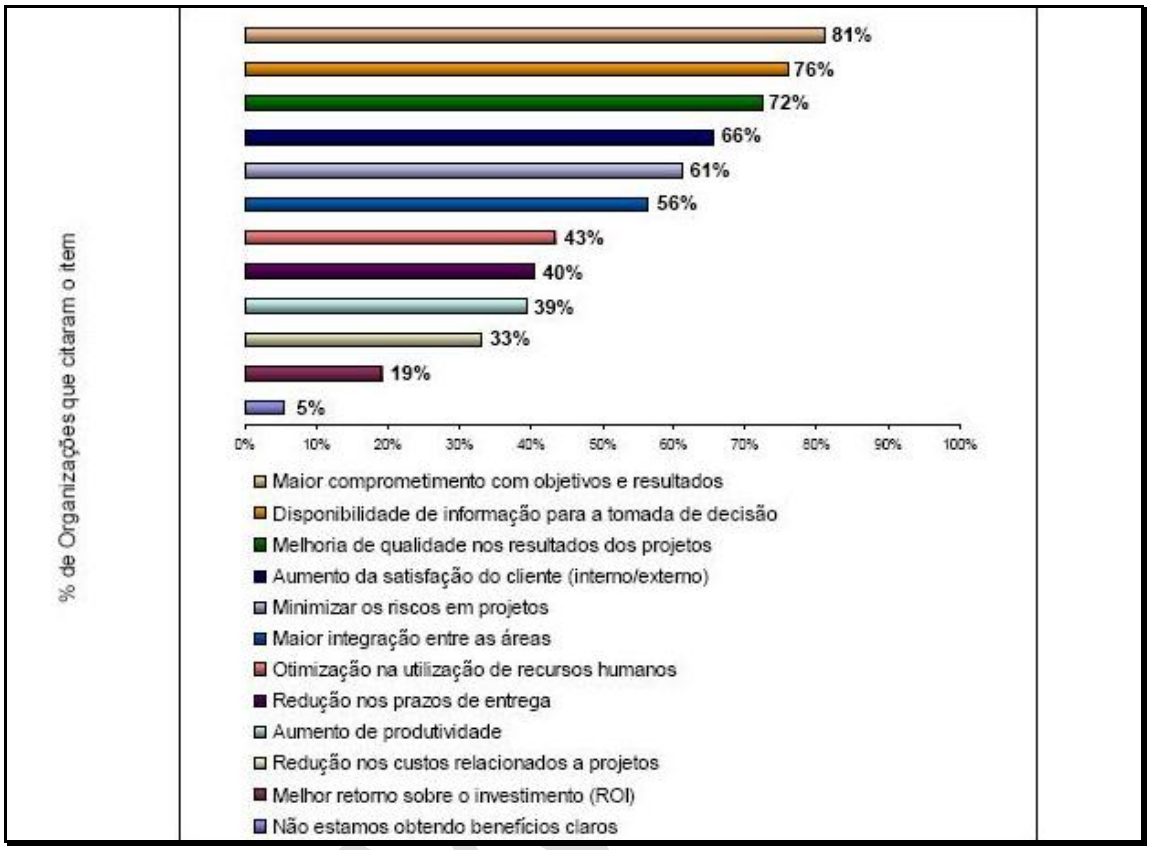


Gráfico 4: Benefícios Obtidos por Empresas com Gerenciamento de Projetos

Fonte: Estudo de Benchmarking em Gerenciamento de Projetos Brasil, PMI – Chapters Brasileiros, 2007

1.2 Questões Chave

A gerência de projetos de software defronta-se com diversos questões cruciais, tanto de ordem técnica quanto gerencial, com maior ou menor intensidade variando conforme o tamanho do projeto ou produto do projeto, as quais desafiam a condução bem sucedida de um projeto de software, tais como:

- Por que se leva tanto tempo para ser ter os sistemas de software concluídos?
- Por que os custos de desenvolvimento são tão elevados?
- Por que existem dificuldades na medição do progresso do desenvolvimento à medida que o software vai sendo construído?
- Como superar as dificuldades na estimação da complexidade dos projetos?
- Como obter a melhoria de produtividade entre desenvolvedores de software?
- Como trabalhar de forma mais inteligente balanceando eficiência e eficácia?
- Como obter a melhoria viável e necessária para controle de progresso dos projetos de software?

Desenvolver, documentar, divulgar, regulamentar, treinar, executar, supervisionar, mudar e aprimorar mecanismos, padrões e procedimentos organizacionais próprios para resolução de questões como mencionadas anteriormente, não são tarefas triviais e, sim, ainda mais desafiadoras. Na realidade, tais tarefas demandam custos de adoção e maturação, além de, requererem boa bagagem de conhecimento técnico e gerencial do pessoal envolvido em tal iniciativa, visando evitar que a utilização de normas corporativas não elevem ainda mais os problemas enfrentados no ciclo de produção de um produto de software, reconhece Rocha (2001).

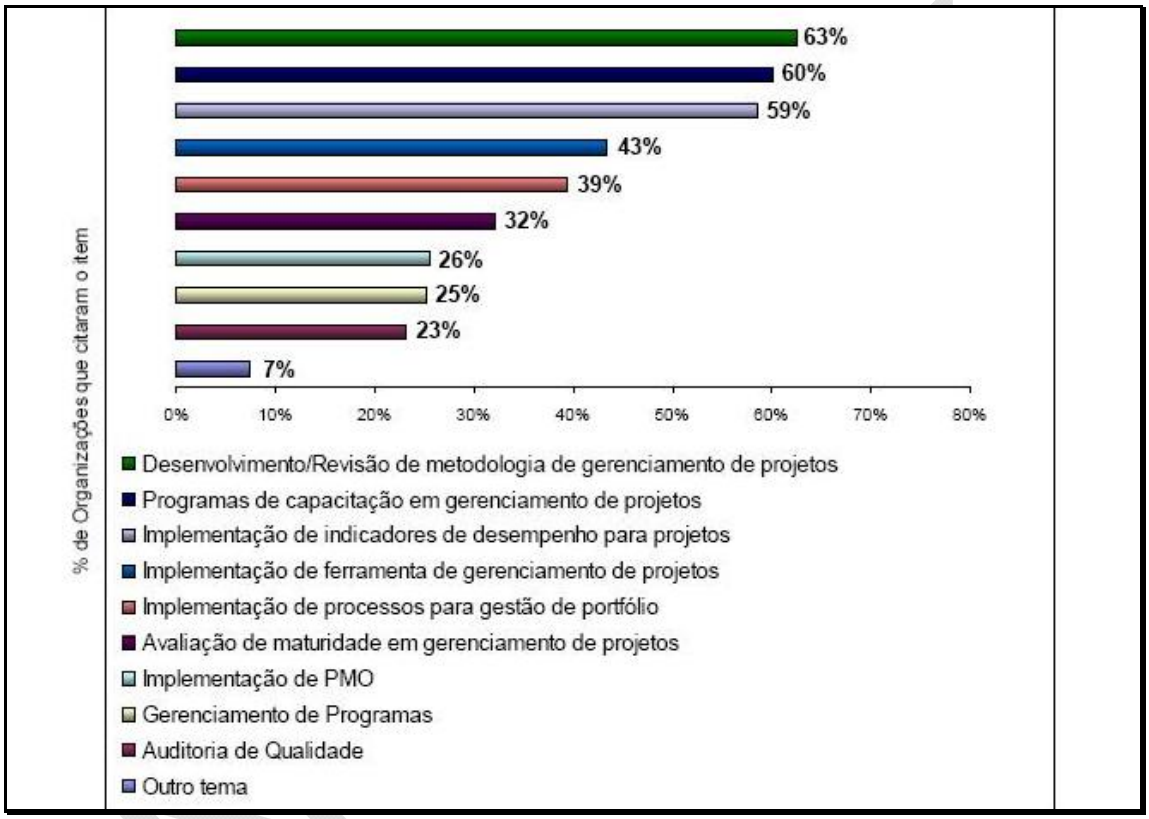


Gráfico 5: Iniciativas das Organizações em Gerenciamento de Projetos

Fonte: Estudo de Benchmarking em Gerenciamento de Projetos Brasil, PMI – Chapters Brasileiros, 2007

Pfleeger (2003) constata que, quando da geração de valor na sociedade do conhecimento, a avidez por inovação demandada pelo mercado e a pressão que o mesmo exerce para que as empresas lancem seus produtos o mais rápido possível, antes que seus concorrentes o façam – caso contrário, a própria viabilidade do negócio pode estar em risco – tendem a complicar ainda mais o cenário de dificuldades para que as organizações consigam estabelecer mecanismos de controle capazes de garantir o êxito em seus empreendimentos de construção de software. Assim, neste ambiente de negócios, as organizações começam a considerar como estratégico para a sua sobrevivência a aquisição, o desenvolvimento e a gestão de novos conhecimentos, bem como, o exercício da criatividade, da flexibilidade e da

facilidade de adaptação para viabilizar respostas com maior rapidez frente às mudanças ambientais – afinal, atualmente, a velocidade é fator tão crucial quanto a qualidade e o custo, conclui Rochar (2001).

Portanto, este estudo acadêmico identifica como oportunidade de contribuição destacar conhecimentos para que as empresas de construção de software possam incentivar o desenvolvimento de novos ativos de processos organizacionais, com enfoque em práticas e procedimentos que possam auxiliar o gerente de projetos de software no desempenho de sua função de controlar o projeto, possibilitando resolver as dificuldades em medir o progresso dos trabalhos e corrigir os rumos do projeto, o plano de projeto ou o processo produtivo, quando necessário.

A estratégia de desenvolvimento neste estudo consiste em explorar e combinar conhecidas técnicas e métodos em uso na indústria de software, tais como, PMBOK, SWEBOK, CMMI, MPS.BR, APF, ISO 12207, ISO 15939, PDCA, BSC, PSM, EVA, AMI, PCU, GQM, reunindo as boas práticas e propondo a adoção de alguns procedimentos essenciais, sistemáticos e pragmáticos, objetivando contribuir para a melhoria da qualidade da gerência de projetos de construção de software, com enfoque na medição e avaliação de desempenho. O enfoque neste trabalho acadêmico consiste em manter a unidade e coerência, como também, o caráter pragmático e efetivo, reunidos em um texto claro, objetivo e sucinto, a despeito da miríade de metodologias analisadas e relacionadas a gestão, a produção e a aquisição de software.

1.3 Estado da Arte

Na visão de Pfleeger (2003), o aprimoramento é a força motriz das pesquisas em engenharia de software com intuito de aprimorar processos, métodos, ferramentas, padrões e práticas de modo a construir e manter produtos de software melhores. Neste sentido, a medição de software tem se tornado uma atividade crucial para a boa prática de gerência de engenharia de software. Quantificando onde e o que podemos aprimorar, descrevemos nossas ações e suas consequências em uma linguagem matemática,o que nos permite avaliar nosso progresso, defende Pfeeger (2003). Ademais, uma abordagem quantitativa permite compararmos o progresso entre diferentes projetos.

Para Pfleeger (2003), a medição pode ajudar a tornar mais visíveis características específicas de processos e produtos, possibilitando transformar a compreensão no mundo real e empírico para elementos e relações no mundo formal da matemática, onde podemos manipular as relações para entendê-las melhor. Nesta abordagem, utilizamos a matemática para resolver um problema, procurar tendências ou caracterizar uma situação. O modelo matemático pode ser aplicado como parte da

solução de um problema empírico que estamos tentando resolver. A Figura 1 ilustra este raciocínio.

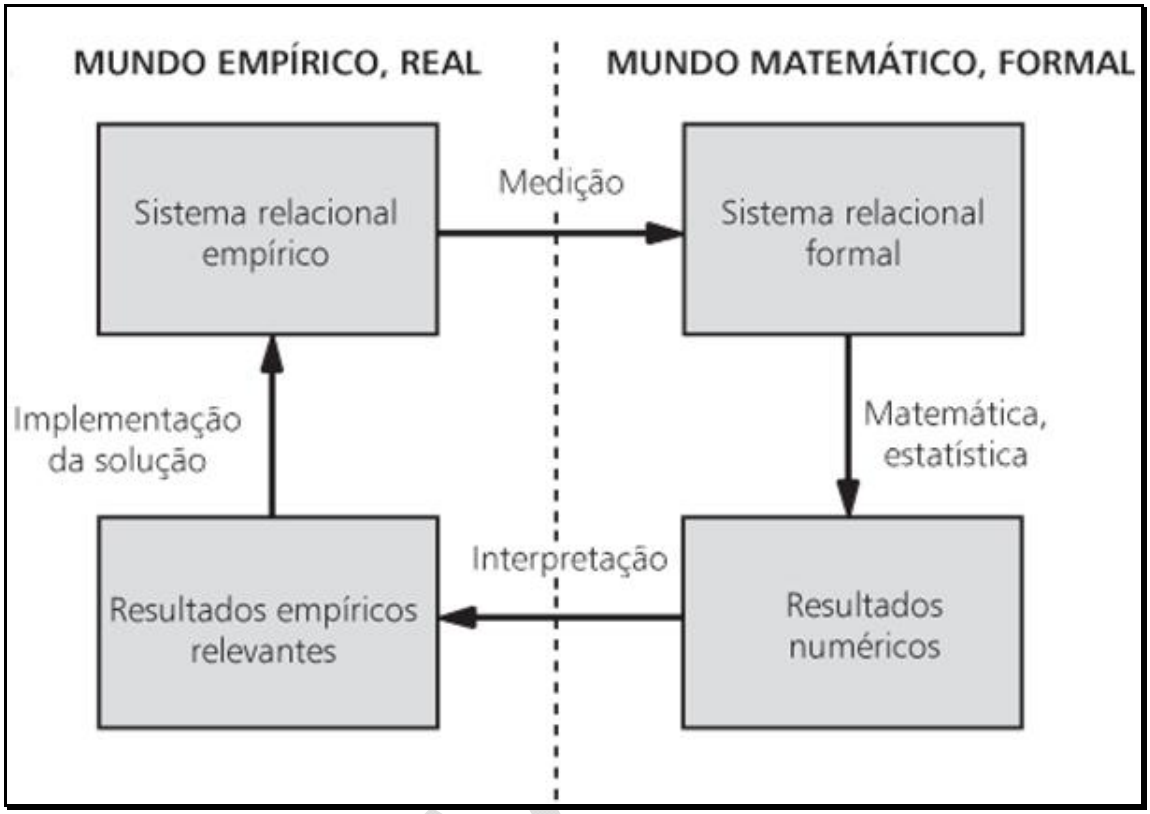


Figura 1: Utilizando medição para encontrar uma solução

Fonte: Pfleeger, 2003

A medição de software se ocupa em obter um valor numérico para alguns atributos de um produto ou processo de software, afirma Sommerville (2007). Ainda destacando esta perspectiva, as medições de software podem ser utilizadas para coletar dados quantitativos sobre o software e seu processo. Uma métrica de software é qualquer tipo de medida que se refira a um sistema de software, processo ou documentação associada. Os valores das métricas de software, que são coletadas, podem ser utilizados para fazer inferências sobre a qualidade do produto de software ou dos processos de software. A Figura 2 ilustra atividades essenciais para se desenvolver um processo de medição de software. Não existem métricas de software padronizadas e universalmente aplicáveis. As corporações desenvolvedoras de software precisam selecionar suas métricas e analisar as medições feitas baseadas no conhecimento e experiências locais, propõe Sommerville (2007).

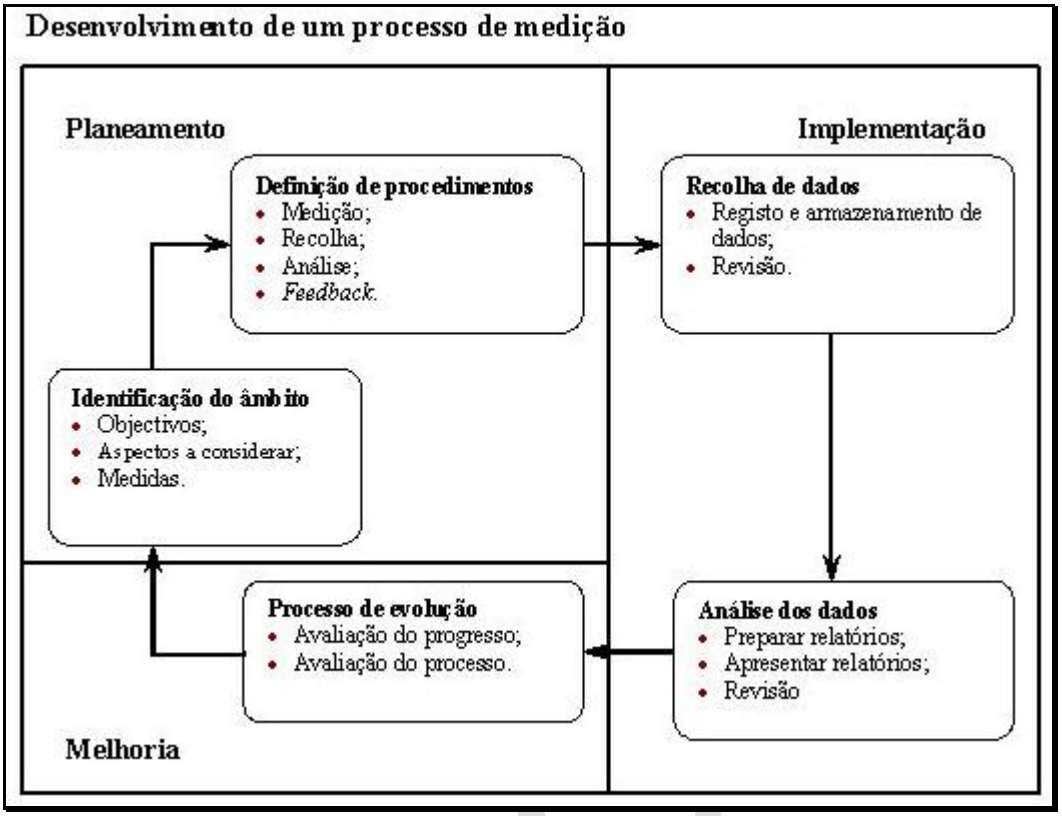


Figura 2: Atividades essenciais para desenvolver um processo de medição

Fonte: Vazquez, 2006

A norma MPS.BR (2007) define que o processo de medição deve ser capaz de coletar, analisar e relatar os dados relativos aos produtos produzidos e aos processos operacionais na organização e em seus projetos, de forma a apoiar os objetivos organizacionais, contemplando os seguintes resultados esperados:

- Objetivos de medição são estabelecidos e mantidos a partir dos objetivos da organização e das necessidades de informação de processos técnicos e gerenciais;
- Um conjunto adequado de medidas, orientado pelos objetivos de medição, é identificado e/ou definido, priorizado, documentado, revisado e atualizado;
- Os procedimentos para a coleta e o armazenamento de medidas são especificados;
- Os procedimentos para a análise da medição realizada são especificados;
- Os dados requeridos são coletados e analisados;
- Os dados e os resultados de análises são armazenados;
- As informações produzidas são usadas para apoiar decisões e para fornecer uma base objetiva para comunicação aos interessados.

2 MÉTRICAS DE PRODUTO

2.1 Conceitos Fundamentais

Uma métrica de software é qualquer tipo de medida que se refira a um sistema de software, processo ou documentação associada, destaca Sommerville (2007). As métricas podem ser preditivas ou de controle. As métricas de controle são associadas a processos de software. As métricas preditivas são associadas a produtos de software. Exemplos de métricas de controle são o esforço e o tempo médio para reparar defeitos no produto. Exemplos de métricas preditivas incluem tamanho do código, facilidade de uso de um sistema de software. Tanto a métrica de controle como a preditiva podem influenciar na tomada de decisões em gerenciamento, como ilustra a Figura 3, conclui Sommerville (2007).

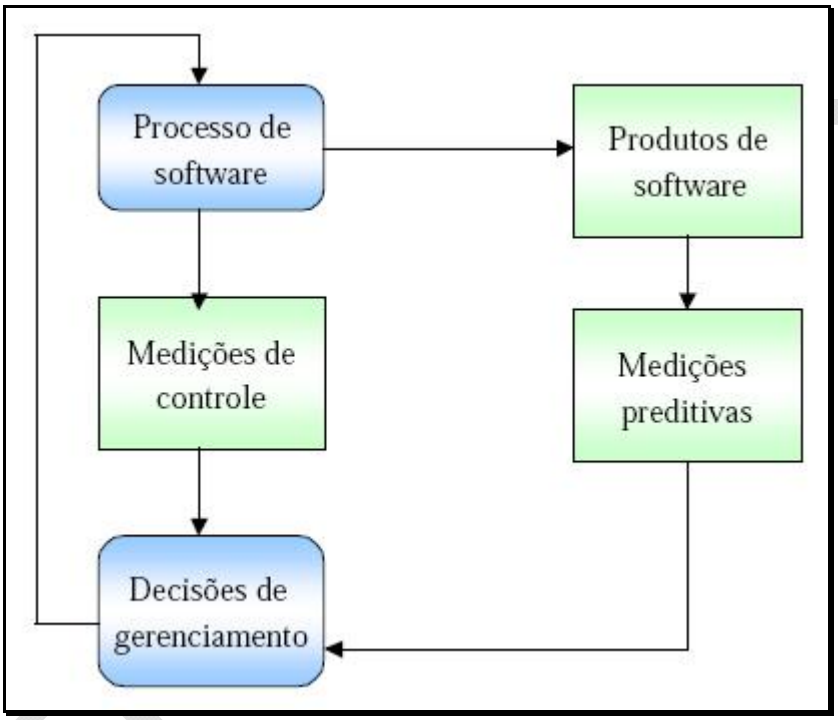


Figura 3: Métricas de Controle e Preditivas

Fonte: Sommerville, 2007

2.2 Importância Estratégica

Uma das atividades mais cruciais para um gerente de projeto de software seja a elaboração de estimativas de esforço, prazo e custo de um projeto de construção de software. Para suportar a realização de tal atividade torna-se necessário a adoção de um método de estimativa de tamanho de sistemas de software.

Segundo Vazquez (2006), a preocupação com aplicação de medições na condução de projetos de software está associado aos seguintes aspectos:

- Atingir o prazo inicialmente previsto;
- Atingir o orçamento inicialmente previsto;

- Geração de um produto de software de boa qualidade, adequado ao uso
- Satisfação do cliente / usuário;
- Fornecimento de informações à gerência de desenvolvimento para que possa melhorar, continuamente, os processos de planejamento, desenvolvimento de software e gestão do produto;
- Motivar equipe;
- Direcionar trabalhos – prover informações a tempo de afetar o próprio processo;
- Identificar oportunidades de melhorias de processo – medir impacto de técnicas e ferramentas;
- Prover a gerência de projetos de indicadores – avaliar o ambiente de forma, a saber, se estamos no caminho certo.

Para tanto, é preciso controlar e supervisionar o processo produtivo, visando manter a produtividade nos níveis previstos, remover o quanto antes defeitos introduzidos no produto, reduzindo ou eliminando o esforço de retrabalho, consequentemente mantendo o orçamento sob controle.

2.3 Pontos de Função

O processo de cálculo do tamanho de um sistema de software utilizando a técnica de pontos de função envolve as seguintes etapas:

1) Determinar o tipo de contagem de pontos de função
2) Identificar o escopo da contagem e a fronteira da aplicação
3) Identificação das funções do sistema para determinar qual a contribuição destas funções à contagem de pontos de função não ajustados.
4) Classificação de cada função quanto à complexidade funcional relativa como: simples, média ou complexa.
5) Cálculo dos pontos de função brutos através da aplicação dos pesos de acordo com a tabela específica
6) Avaliação das 14 características gerais do sistema
7) Determinação do Fator de Ajuste
8) Cálculo dos pontos de função ajustados.

1. 3.1 Etapas da Parte I - Levantamento dos Pontos de Função Brutos.

A técnica de pontos de função pode ser aplicada para diversos tipos de necessidade. Levando-se em conta a necessidade de estimativas mais precisas e de melhoria no gerenciamento de projetos de desenvolvimento de sistemas, a TPF pode e deve ser aplicada desde as primeiras fases do projeto e recomenda-se a sua inclusão no ciclo de desenvolvimento de sistemas. Para iniciar a contagem de pontos de função temos que

determinar o tipo de contagem que será aplicado, o escopo e a fronteira da aplicação considerados.

3.1.1 Determinar o tipo de contagem de pontos de função

A determinação do tipo de contagem de pontos de função deve ser realizada em função do objeto da contagem, que pode ser dirigido à projetos de desenvolvimento, de manutenção ou aplicações prontas.

3.1.1.1 Contagem em Projetos de Desenvolvimento

A contagem de projetos de desenvolvimento visa medir as funções oferecidas para o usuário quando o software é instalado pela primeira vez para o usuário. Esta contagem deve ser aplicada para medir as funcionalidades que serão disponibilizadas por algum novo desenvolvimento. O projeto de desenvolvimento pode incluir contagem de parte de conversão de dados.

3.1.1.2 Contagem em Projetos de Manutenção

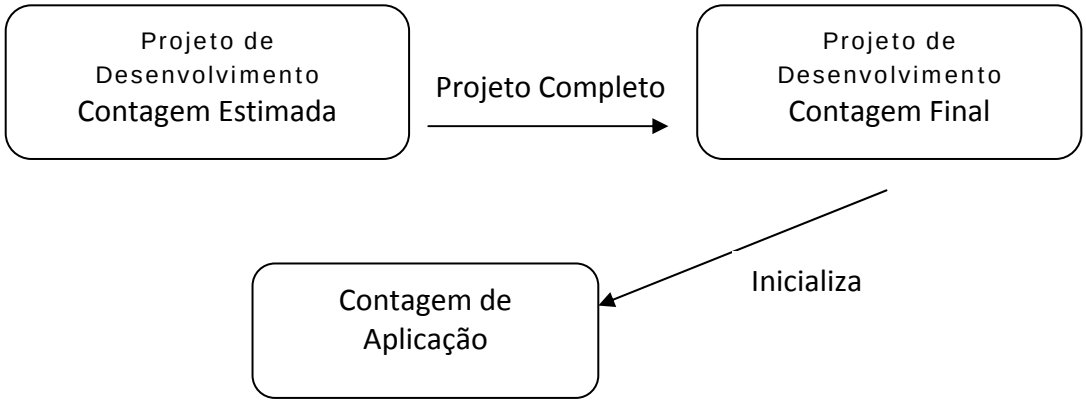
A contagem de projetos de manutenção visa medir as modificações de uma aplicação existente que incluem, alteram ou excluem funcionalidades disponibilizadas para o usuário quando o projeto for concluído.
Quando a funcionalidade originada de um projeto de manutenção for implantada, a contagem de pontos de função da aplicação instalada deverá ser atualizada para refletir as alterações na funcionalidade da aplicação. O projeto de manutenção pode incluir contagem de parte de conversão de dados.

3.1.1.3 Contagem de Aplicações

A contagem de aplicações visa medir a funcionalidade de um software instalado. Este número é inicializado quando o projeto de desenvolvimento é concluído e deve ser atualizado constantemente a cada nova versão do software, de forma que esta contagem forneça uma medida das funções correntes que o software provê para o usuário.

Diagrama de Tipos de Contagem de Pontos de Função

O diagrama a seguir ilustra os tipos de contagem de pontos de função e o relacionamento entre eles.



3.1.2 Identificar o Escopo da Contagem e a Fronteira da Aplicação

O escopo da contagem define a funcionalidade que deve ser incluída na contagem de pontos de função.

A fronteira da aplicação é um limite conceitual que serve para separar o software que está sendo medido dos usuários. O usuário pode ser qualquer pessoa que defina os requisitos funcionais do software e/ou qualquer pessoa ou coisa que se comunique ou interaja com o software, inclusive outro sistema.

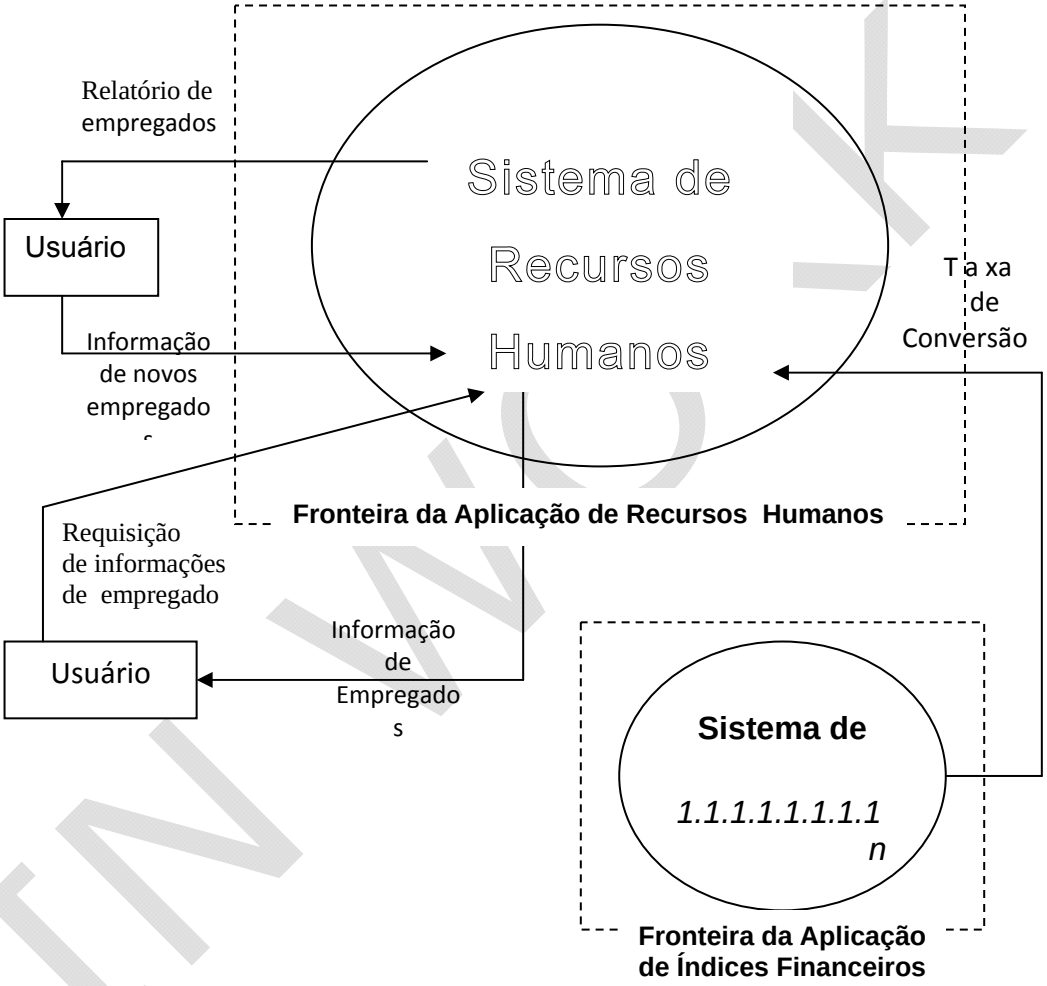


Figura 3.1.2 - Ilustração das Fronteiras de Aplicações

3.1.3 Introdução da etapas necessárias para determinar a quantidade de pontos de função não ajustados.

A quantidade de pontos de função não ajustados será determinada a partir de pesos previamente estabelecidos para cada tipo de função definida. Para que estes pesos sejam corretamente atribuídos, é necessário identificar a natureza de cada elemento da contagem.
A figura a seguir ilustra as etapas necessárias para o cálculo dos pontos de função brutos. A descrição detalhada da forma de cálculo da Parte I será apresentada mais adiante no tópico "4.1 Pontos de Função Brutos".

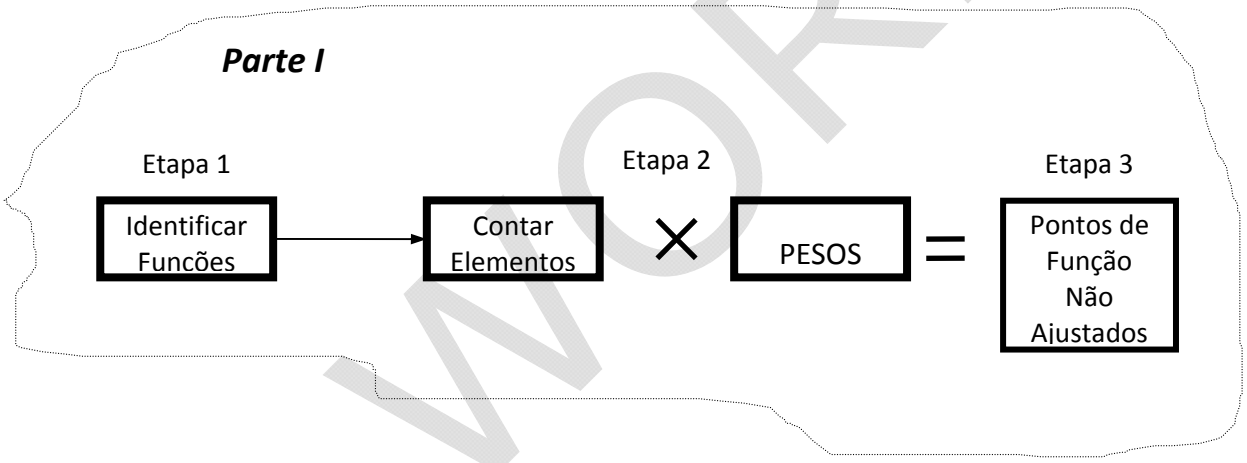


Figura 3.1.3 - Etapas da parte I da TPF - Levantamento dos Pontos de Função

3.1.4 Etapa de Identificação dos elementos da Contagem de Pontos de Função.

Após estabelecer a fronteira da aplicação e o tipo de contagem, o processo de identificação consiste no levantamento das funções relevantes para estimativa do cálculo do tamanho do sistema. Utilizando TPF para projeto de desenvolvimento de sistemas, na fase inicial de seu ciclo de vida, onde não se tem todas as informações do sistema, deve-se extrair alguns elementos básicos que permitam a contagem dos pontos de função brutos, como as entidades do modelo preliminar de dados, as funções definidas no diagrama de contexto e no diagrama de fluxo de dados, por exemplo. Nesta etapa serão identificados os arquivos lógicos internos e arquivos de interfaces, entradas externas, saídas externas e consultas externas do sistema.

3.1.5 Etapa de Classificação de cada função quanto à Complexidade Funcional

Após a identificação dos elementos de contagem – arquivos lógicos internos, arquivos de interface, entradas externas, saídas externas e consultas externas- deve-se classificar cada um dos elementos segundo a complexidade.
Cada elemento será classificado em três níveis de complexidade – simples, médio ou complexo.
Os elementos de dados terão a complexidade analisada de acordo com a quantidade de tipos de registros e de campos de dados.
Os elementos de transação serão avaliados em função da quantidade de arquivos lógicos internos ou de interface referenciados e também em função da quantidade de campos de dados.

3.1.6 Cálculo dos pontos de função brutos através da aplicação dos pesos de acordo com a tabela específica

Para cálculo da quantidade de pontos de função não ajustados, cada função identificada na etapa anterior e classificada de acordo com o seu nível de complexidade, receberá um peso equivalente à quantidade de pontos de função que cada elemento contribui para o tamanho da aplicação.

A tabela abaixo, auxilia a visualização desta contagem.

VARIÁVEIS	QUANTIDADE		PESO		TOTAL
- ARQUIVOS		X		=	
- INTERFACES		X		=	
- ENTRADAS		X		=	
- SAÍDAS		X		=	
- CONSULTAS		X		=	
TOTAL DO SOMATÓRIO					

Figura 3.1.6 - Análise de Funções para cálculo dos pontos de função brutos

2. 3.2 Etapas da Parte II - Levantamento dos pontos de função ajustados.

O cálculo do fator de ajuste é baseado em quatorze características gerais do sistema. Cada característica está associada a descrições que auxiliam a determinação do nível de influência de cada uma. As características gerais de um sistema podem influir no seu tamanho de -35% a +35%. A etapa 4 consiste da pontuação do nível de influência baseado nas quatorze característica gerais do sistema e a etapa 5 na determinação do fator de ajuste que poder variar de 0,65 a 1,35.

Para calcular o fator de ajuste deve-se seguir os seguinte passos:

* Avaliar o impacto de cada uma das quatorze características gerais no sistema que está sendo contado, atribuindo para cada característica um peso de 0 a 5.
* Calcular o nível de influência através da soma dos pesos de cada uma das características.
* Calcular o fator de ajuste a partir da equação Fator = (NI * 0,01) + 0,65.

Exemplificando, se a soma for pesos das quatorze características for 61, o cálculo do fator de ajuste deverá ser:

Fator = (61 * 0,01) + 0,65 = 1,26

Uma vez que se possua o valor do fator de ajuste, a fórmula de cálculo dos pontos de função(ajustados) de um projeto de desenvolvimento é dada por:

Pontos de função = (Pontos de função brutos X Fator de ajuste).

Utilizando o fator de ajuste do exemplo anterior, podemos calcular os pontos de função de um projeto de desenvolvimento da seguinte forma:

Pontos de Função (Ajustados) = Pontos de Função Brutos X Fator de ajuste
= 200 X 1,26
= 252 pontos

A descrição detalhada da forma de obtenção dos pontos de função ajustados será apresentada no tópico "4.2 Ajuste dos Pontos de Função Brutos".

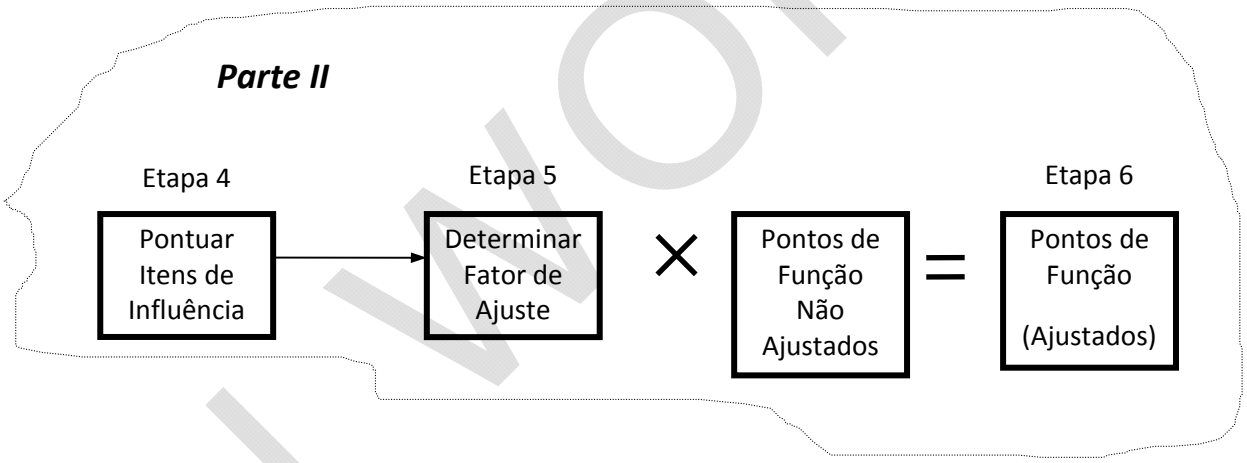


Figura 3.2 - Etapas da parte II da TPF - Análise de Influência

3. 3.3 Análise de Recursos e Prazos a Partir do Cálculo de Pontos de Função

Considerando-se a seguinte projeção para produtividade por fase da Metodologia de Desenvolvimento de Sistemas:

Fase	Percentual de Esforço	Produtividade em HS / PF
Análise	20 %	15
Especificação	20 %	15
Construção	30 %	07
Teste	15 %	07
Implantação	5 %	15

Considerando-se a seguinte alocação efetiva de recursos por fase da ASC:
Recursos para a fase de análise: 3 homens-mês
Recursos para a fase de especificação: 4 homens-mês
Recursos para as fases de construção, teste e implementação: 5 homens-mês

Seja FPA = Total de Pontos de Função Ajustado contabilizado na etapa II.

Fase de Análise = FPA X 20% ⟶ ____ FPT[1]
Necessidade de homens-mês = ____ X 25 / 160 ⟶ ____ HM
Prazo para a fase = ____ / 3 ⟶ ____(A) MÊS

Fase de Especificação = FPA X 20% ⟶ ____ FPT
Necessidade de homens-mês = ____ X 25 / 160 ⟶ ____ HM
Prazo para a fase = ____ / 4 ⟶ ____(B) MÊS

Fase de Construção = FPA X 30% ⟶ ____ FPT
Necessidade de homens-mês = ____ X 40 / 160 ⟶ ____ HM
Prazo para a fase = ____ / 5 ⟶ ____(C) MÊS

Fase de Teste = FPA X 15% ⟶ ____ FPT
Necessidade de homens-mês = ____ X 15 / 160 ⟶ ____ HM
Prazo para a fase = ____ / 5 ⟶ ____(D) MÊS

Fase de Implantação = FPA X 5% ⟶ ____ FPT
Necessidade de homens-mês = ____ X 20 / 160 ⟶ ____ HM
Prazo para a fase = ____ / 5 ⟶ ____(E) MÊS

Prazo para análise e especificação (____(A) + ____(B)) ⟶ ___ Meses

Prazo p/ construção, teste e implantação(C) + (D) + (E) ⟶ ___ Meses

Prazo total (A) + (B) + (C) + (D) + (E) ⟶ ___ Meses

[1] Quantidade de Pontos de Função da fase de análise.

2.4 Pontos de Caso de Uso

Os Pontos de Casos de Uso (PCU) foram criados em 1993 por Gustav Karner da empresa Objectory AB. Esta métrica permite fazer estimativas no início do projeto com base no modelo de casos de uso. A filosofia dos Pontos de Casos de Uso é baseada na definição da Análise de Pontos de Função (APF), na qual a funcionalidade vista pelo usuário é a base para a estimativa do tamanho do software.

O procedimento de contagem de casos de uso consiste nos seguintes passos (MEDEIROS, 2004):

9) Relacionar os atores, classificá-los de acordo com seu nível de complexidade (simples, médio ou complexo) atribuindo respectivamente os pesos 1, 2 ou 3, conforme Quadro 1. Calcule o TPNAA (Total de Pesos não Ajustados dos Atores) somando os produtos da quantidade de atores pelo seu peso.

Quadro 1: Classificação de Atores

Complexidade do Ator	**Descrição**	**Peso**
Simples	Muito poucas entidades de Banco de Dados envolvidas e sem regras de negócio complexas	1
Médio	Poucas entidades de Banco de Dados envolvidas e com algumas regras de negócio complexas	2
Complexo	Regras de negócios complexas e muitasentidades de Bancos de Dados presentes	3

Fonte: Medeiros, 2004

10) Contar os casos de uso e atribuir o grau de complexidade sendo a complexidade baseada no número de classes e transações. Calcule o TPNAUC (Total de Pesos não ajustados dos casos de usos) somando os produtos da quantidade de casos de usos pelo respectivo peso conforme o Quadro 2.

Quadro 2: Classificação dos Casos de Uso

Tipo de Caso de	**Descrição**	**Peso**

Uso		
Simples	Considerar até 3 transações com menos de 5 classes de análise (negócio)	5
Médio	Considerar de 4 a 7 transações com 5 a 10 classes de análise	10
Complexo	Considerar de 7 transações com pelo menos de 10 classes de análise	15

Fonte: Medeiros, 2004

11) Calcular PCU´s não ajustados, também chamados de PCUNA, de acordo com a seguinte fórmula:

PCUNA = TPNAA+ TPNAUC

12) Determinar o fator de complexidade técnica. Os fatores de complexidade técnica variam numa escala de 0 a 5, de acordo com o grau de dificuldade do sistema a ser construído. O valor 0 indica que a grau não está presente ou não é influente, 3 influência média e o valor 5 indica influência significativa através de todo o processo. Após determinar o valor dos fatores, multiplicar pelo respectivo peso ilustrado no Quadro 3, somar o total e aplicar a seguinte fórmula:

Fator de complexidade técnica (FCT) = 0.6 + (0.01 * Somatório do Fator técnico)

Quadro 3: Fatores de complexidade técnica

Descrição	Peso
Sistemas Distribuídos	2,0
Desempenho da aplicação	1,0
Eficiência do usuário final (on-line)	1,0
Processamento interno complexo	1,0
Reusabilidade do código em outras aplicações	1,0
Facilidade de instalação	0,5
Usabilidade (facilidade operacional)	0,5
Portabilidade	2,0
Facilidade de manutenção	1,0

Concorrência	1,0
Características especiais de segurança	1,0
Acesso direto para terceiros	1,0
Facilidades especiais de treinamento	1,0

Fonte: Medeiros, 2004

13) Determinar o fator de complexidade ambiental: os fatores de complexidade ambientais indicam a eficiência do projeto e estão relacionados ao nível de experiência dos profissionais. Esses fatores descritos no Quadro 4 são determinados através da escala de 0 a 5, onde 0 indica baixa experiência, 3 indica média experiência e 5 indica alta experiência. Após determinar o valor de cada fator, multiplicar pelo peso e somar o total dos valores. Em seguida, aplicar a seguinte fórmula:

Fator de complexidade ambiental (FCA) = 1,4 + (-0,03 * Somatório do Fator Ambiental)

14) Calcular os PCU´s ajustados: esse cálculo é realizado com base na multiplicação dos PCU não ajustados, na complexidade técnica e na complexidade ambiental através da seguinte fórmula:

PCUA = PCUNA * Fator de complexidade técnica * Fator de complexidade ambiental

Quadro 4: Fatores de complexidade ambiental

Fator	Descrição	Peso
F1	Familiaridade com o processo de desenvolvimento de software	1,5
F2	Experiência na aplicação	0,5
F3	Experiência com OO, na linguagem e na técnica de desenvolvimento	1,0
F4	Capacidade do líder de análise	0,5
F5	Motivação	1,0
F6	Requisitos estáveis	2,0
F7	Trabalhadores com dedicação parcial	-1,0
F8	Dificuldade da linguagem de programação	-1,0

Fonte: Medeiros, 2004

15) Calcular a estimativa de horas de programação. Karner, o criador da estimativa, sugere a utilização de 20 pessoas-hora por unidade de PCU. Schneider e Winters (2001) sugerem o seguinte refinamento:

X = total de ítens de F1 a F6 com pontuação abaixo de 3

Y = total de ítens de F7 a F8 com pontuação acima de 3

Se X + Y <= 2, usar 20 como unidade de homens/hora

Se X + Y = 3 ou X + Y = 4, usar 28 como unidade de homens/hora

Se X + Y >= 5, deve-se tentar modificar o projeto de forma a baixar o número, pois risco de insucesso é relativamente alto.

Estimativa de horas = PCUA * pessoas hora por unidade de PCU

2.5 Pontos de Objeto

2.6 KLOC

2.7 COCOMO II

Incluir texto técnico.

3 PROCESSO DE MEDIÇÃO

3.1 Importância Estratégica

De acordo com Pfleeger (2003), organizações desenvolvedoras de software deveriam adquirir capacidades para medir e avaliar os produtos de software e o modo como são produzidos. As técnicas de medição disponíveis propõem um núcleo comum de atividades, ou seja: medimos os aspectos principais de produtos, processos e recursos, utilizamos essas informações para avaliar se estamos satisfazendo os objetivos de produtividade, desempenho, qualidade e outras características desejáveis do projeto do produto de software.

Segundo Demarco (1982), a necessidade de se medir em engenharia de software pode ser resumida no seguinte pensamento: “não se pode controlar aquilo que não se consegue medir”. Por outro lado, uma boa cultura de gerenciamento de projetos de software supõe a possibilidade de predizer o comportamento futuro dos produtos e processos de software, sendo necessário contar com dados apropriados e confiáveis. Nesses casos a medição – processo cujo principal foco é apoiar a tomada de decisão em relação aos projetos, processos e atendimento aos objetivos organizacionais – torna-se importante, uma vez que “não se pode predizer o que não se pode medir” (FENTON & PFLEEGER, 1997).

Segundo Pressmann (2002), a medição permite aos gerentes planejar, controlar, melhorar e aperfeiçoar o processo de desenvolvimento de software. Medição resulta em mudança cultural. Coletar dados, calcular e analisar métricas são atividades que devem ser implementadas para iniciar um programa de medições. O custo e o esforço necessários para construir software – mesmo em linhas de código produzidas – são fáceis de coletar, desde que convenções específicas para a medição sejam estabelecidas antecipadamente.

Sommerville (2007) destaca que, o uso de medição e de métricas sistemáticas de software ainda é relativamente incomum. As organizações de software relutem em introduzir programas de medição, pois os benefícios não são bem definidos. Essa relutância pode ser explicada devido ao fato que, em muitas empresas, os processos produtivos utilizados nos projetos de software ainda são organizados de maneira precária e não estão suficientemente aperfeiçoados para fazer uso de medições. Outra razão é que não há padrões para as métricas, decorrendo em suporte ferramental limitado para realização de coleta e análise de dados. As organizações permanecerão despreparadas para implantar medições até que esses padrões e ferramentas de métricas estejam disponíveis, ressalta Sommerville (2007).

Rocha (2001) comenta que, com intuito de aperfeiçoar a construção de software e obter produtos mais confiáveis e com qualidade desejada, intensificou-se a pesquisa

sobre o processo produtivo de software e, neste sentido, várias normas e modelos de qualidade foram definidos para auxiliar as organizações na definição e melhoria de processos. Rochar (2001) ressalta que, para possibilitar aprimoramento de cada fase do ciclo de produção é necessário a obtenção e análise de dados quantitativos capazes de revelar a realidade do processo produtivo. Neste sentido, a realização de medições é reconhecida como premissa essencial para a melhor prática da gerência de engenharia de software.

Rocha (2001) comenta que, programas de medição e métricas de software foram propostos e aplicados na prática visando alcançar os objetivos a seguir:

- Melhorar o entendimento sobre o processo, produto, recursos e ambiente de desenvolvimento e, assim, estabelecer bases para comparação entre as medições;
- Avaliar o progresso do projeto comparando com dados planejados;
- Fazer estimativas sobre o futuro andamento do projeto com base em comportamentos passados;
- Promover melhorias pela identificação de falhas, ineficiências e outras oportunidades para melhorar a qualidade do produto e seu processo de produção.

Entretando, definir, coletar e analisar um conjunto de métricas de software não é tarefa trivial, afirma Rocha (2001). Medição de software requer conhecimento especializado para evitar que seu uso não resulte em aumento dos custos e problemas na construção de software. Basili e Rombach (1994) ressaltam que, um programa de medições de software para resultar em benefícios deve abordar os seguintes aspectos:

- Orientar-se em objetivos especifícos;
- Ser aplicado sobre todos os produtos, processos e recursos do ciclo de vida do projeto;
- Ter seus resultados interpretados com base em características do contexto organizacional e do ambiente de projeto.

Segundo a norma NBR ISO/IEC 12207 (1998), o propósito da medição é coletar e analisar os dados relativos aos produtos desenvolvidos e aos processos implementados na organização e em seus projetos, de forma a apoiar o efetivo gerenciamento dos processos e demonstrar objetivamente a qualidade dos produtos.

A norma MPS.BR (2007) ressalta que, um processo de medição deve responder principalmente às seguintes questões:

- Que valor esta medição vai agregar para aqueles que forneceram os dados e para os que irão receber a análise dos resultados?
- Essas medições são úteis para os que coletam e utilizam os dados?

Medição é um dos processos principais para gerenciar as atividades do ciclo de vida do projeto de software e avaliar a viabilidade dos planos de projeto, destaca a norma MPS.BR (2007).

Premissas a considerar num processo de medição, de acordo com método GQM (BASILI & ROMBACH, 1994):

- Prover resultados consistentes;
- Permitir sua realização por pessoas não especialistas;
- Ser de fácil aprendizado;
- Ser compreensível ao usuário final;
- Servir para estimativas;
- Permitir automatização;
- Possibilitar obtenção de séries históricas.

3.2 Processo de Medição do Produto

Um processo genérico de medição de software, conforme proposto por Sommerville (2007), ilustrado pela Figura 4, pode ser especificado nas atividades descritas abaixo:

- Escolha de medições a serem feitas: Devem ser formuladas as questões às quais as medições se destinam a responder, e as medições exigidas para responder a essas questões devem ser definidas. As medições que não foram diretamente relevantes para essas questões devem ser definidas. O método GQM criado por Basili e Rombach (1994) é uma boa abordagem a ser utilizada quando se deseja definir que medidas devem ser coletadas.
- Seleção de artefatos de software a serem avaliados: Pode não ser necessário ou desejável avaliar valores de métricas para todos os artefatos em um sistema de software. Uma seleção representativa de componentes pode ser escolhida para a medição. De outro modo, podem ser avaliados somente os artefatos cruciais, que estão em uso constatemente.
- Medição de atributos dos artefatos de software: Os artefatos selecionados são medidos e os valores das medidas são computados. Isso envolve processar a representação dos artefatos (código, projeto, documentação) utilizando uma ferramenta automatizada de coleta de dados.
- Identificação de medições anômalas: Uma vez realizada as medições de artefatos, elas devem ser comparadas entre si e com as medições

precendentes, que tenham sido registradas em um banco de dados de medições. É preciso procurar valores altos ou baixos incomuns para cada métrica, uma vez que isso sugere que pode haver problemas com o artefato que apresenta estes valores.

- Análise de artefatos anômalos: Identificados os artefatos com valores anômalos em medidas particulares, os mesmos devem ser inspecionados para decidir se os valores anômalos significam que a qualidade do artefato está comprometida ou não.

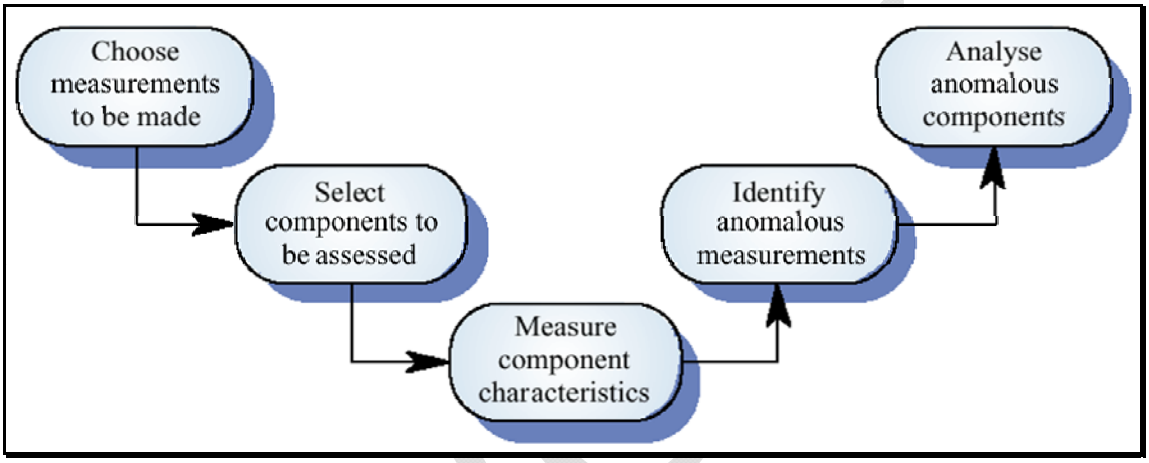


Figura 4: Esboço de processo genérico de medição de software

Fonte: Sommerville, 2007

3.3 Medição Integrada ao Processo Produtivo

A implantação de um programa de medições requer um esforço específico da organização. Requer um compromisso muito forte em enfrentar todos as dificuldades que surgem durante esta longa trajetória sem fim. Não existe espaço para acomodação, sempre está se renovando e apreendendo com as novas informações que são coletadas.

A norma MPS.BR (2007) define que, a divisão de engenharia de software de uma organizacionação deve demonstrar o seu compromisso com as medições pelo estabelecimento de uma política de medição, o que implica na designação de responsabilidades e de treinamento e na alocação de recursos e orçamento. Para cada responsabilidade definida, devem ser designadas pessoas competentes, que devem saber sobre os conceitos que envolvem a medição, como os dados serão coletados, analisados e comunicados. Dentre os papéis envolvidos, temos: usuário da medição, analista de medição, bibliotecário da medição. Essa definição de papéis não implica que estes sejam assinalados para pessoas diferentes: mais de um papel pode ser executado pela mesma pessoa. O tempo e o esforço requeridos para realizar medições são significativos, sendo necessário direcionar os esforços envolvidos.

A implantação de um programa de medições alinhado a objetivos estratégico da organização, na seqüência lógica de ações, como descrita sucintamente a seguir:

1) Documentar o processo produtivo utilizado e padronizá-lo (sem buscar a realização de melhorias, mas tão somente a estabilização do processo);
2) Estabelecer os objetivos do programa de medições (por exemplo, determinar a produtividade);
3) Definir as métricas necessárias ao alcance dos objetivos pretendidos (por exemplo: tamanho funcional em pontos de função, esforço em homem-horas, qualidade em densidade de erros, etc.);
4) Identificar os dados a serem coletados (por exemplo: horas trabalhadas de cada técnico envolvido, tamanho funcional de cada uma das solicitações de alteração do software ou de cada release entregue ao cliente, erros identificados e suas categorias, etc.);
5) Definir o processo de coleta de dados (identificar os pontos de coleta, periodicidades, formulários ou sistema de coleta, etc.);
6) Obter ferramentas (construir e/ou adquirir as ferramentas que permitam implementar o processo de coleta anteriormente definido);
7) Criar um banco de dados de métricas (implementar um repositório que possa abrigar, de forma organizada e acessível, todos os dados que virão a ser coletados);
8) Definir um mecanismo de feedback (uma maneira que permita à equipe de métricas receber feedback de todos os participantes do processo). Existem várias outras formas de se conduzir o processo de implantação de um programa deste tipo. O importante é o comprometimento da gerência e a clara definição dos objetivos do programa, de modo a garantir que as métricas geradas sejam aquelas necessárias e efetivamente utilizadas.

Uma vez implantado o programa de medições, a organização começará a obter dados de produtividade. Em uma organização imatura, ainda no nível 1 na norma CMMI, é provável que a produtividade varie de forma errática - refletindo, é claro, a realidade que está sendo monitorada. Um processo de medição deve ser implementado de forma evolutiva dentro da organização, pois ele é conseqüência da maturidade dos outros processos corporativos (produtivos, gerenciais). Inicialmente, as medições são difíceis de serem feitas e os dados são difíceis de coletar, como conseqüência de processos ainda imaturos, ressalta a norma MPS.BR (2007). Gradativamente, será possível separar os projetos por plataforma, ramo de negócio, localização geográfica, perfil da equipe e outros fatores que estejam influenciando os resultados. O tratamento de categorias separadas tenderá a reduzir a variabilidade.

Após a estabilização dos fatores mais influentes, será possível estabelecer uma produtividade média para cada categoria definida. Esse será o ponto de partida para a obtenção de melhorias junto aos fornecedores.

4 ANÁLISE DE MEDIÇÕES

4.1 Integração com EVA

4.2 Relatórios de Desempenho

Incluir texto técnico.

5 BOAS PRÁTICAS

5.1 Base Histórica de Medições

Sommerville (2007) reforça a importância para que os dados coletados em um processo de medição devem ser mantidos com um recurso organizacional, e os registros históricos de todos os projetos devem ser mantidos mesmo quando os dados tenham sido utilizados durante um determinado projeto. Uma vez estabelecido um banco de dados de medição suficientemente grande, as comparações entre projetos podem ser efetuadas e métricas específicas podem ser aprimoradas, de acordo com as necessidades organizacionais. A utilização de dados históricos para o planejamento de projetos é fundamental e, a banco de dados de medição é alimentado continuamente com os dados que vão sendo extraídos dos projetos em andamento e dos encerrados.

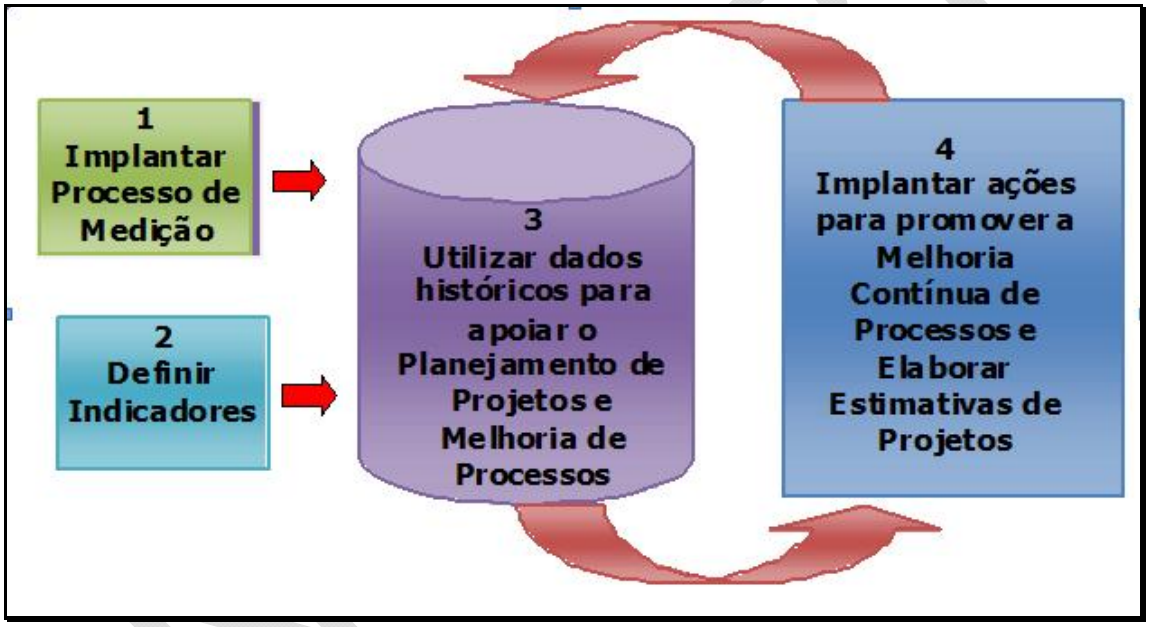


Figura 5: Influência da base de histórica de medições

Fonte: do autor

Rocha (2001) afirma que dados históricos darão subsídios para a criação de indicadores de desempenho financeiro, qualidade e produtividade que retratem a construção de sistemas de software. No esforço de promoção de cultura organizacional voltada para a qualidade e produtividade, não é admissível o planejamento de projeto de software através de intuição (‘feeling’), onde questões como: qual é a produtividade da setor de engenharia de software, qual é a capacidade de produção, qual conjunto de ferramentas possibilita a maior produtividade, quais são os indicadores de qualidade existentes, qual método de construção devemos adotar, desenvolver ou comprar um pacote de software e customizá-lo, são simplesmente

negligenciadas ou respondidas sem suporte de uma base quantificável, gestão do conhecimento e aplicação de metodologias gerenciais.

5.2 Padronização

5.3 Escolha da Métrica

A dificuldade crucial para implantar um processo de medição é determinar o que medir. Basili e Rombach (1994) propuseram o método GQM (goal-question-metric) para auxiliar a definir que medições devem ser feitas e como elas devem ser utilizadas. O método essencialmente é baseado nos seguintes questionamentos:

- Quais são as metas/objetivos?

  O que a empresa está tentando alcançar. Metas podem ser a melhoria de produtividade da equipe de programadores de software, um tempo mais curto de construção do produto de software, um aumento de confiabilidade do produto, etc.
- Quais questões se deseja responder?

  São aperfeiçoamentos das metas, quando são identificadas áreas específicas de incerteza, relacionadas às metas. Normalmente, uma meta terá uma série de questões associadas, que precisam ser respondidas. Exemplos de questões relacionadas às metas mencionadas anteriormente são:
  - Como pode ser aumentado o número de linhas de código depuradas?
  - Como pode ser reduzido o tempo requerido para finalizar os requisitos do produto?
  - Como podem ser feitas avaliações de confiabilidade mais eficazes?
- Quais métricas poderão ajudar?

  São as medidas que precisam ser coletadas para ajudar a responder às questões e para confirmar se as melhorias de processo produtivo alcançaram a meta desejada ou não. Referente aos exemplos anteriores, dentre as medidas que podem ser feitas destacam-se:
  - A produtividade individual dos programadores em linhas de código e em seu nível de experiência;
  - O número de comunicações formais entre cliente e fornecedor para cada mudança de requisito;
  - O número de testes necessários para provocar uma falha no produto.

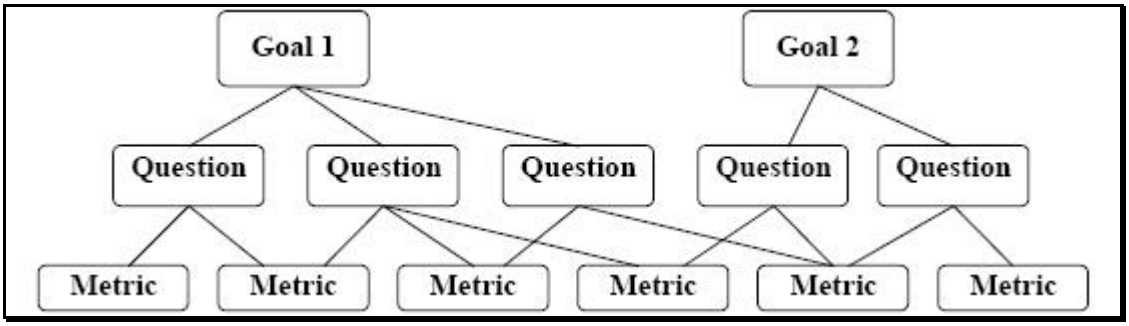


Figura 6: Estrutura hierárquica do método GQM

Fonte: Basili e Rombach, 1994

Resumidamente, as fases do método GQM são detalhadas em atividades como descritas no Quadro 5.

Quadro 5: Fases no método GQM

Fase	Atividade
Planejamento	• Preparar e motivar membros da organização; • Definir objetivos, cronogramas e responsabilidades; • Estabelecer equipe de medição; • Selecionar área de melhoria; • Treinar pessoal envolvido.
Definição	• Definir objetivos, questões e métricas; • Conduzir entrevistas; • Verificar as métricas definidas.
Coleta de Dados	• Executar o Plano de Medição, coletando e armazenando os resultados.
Interpretação	• Analisar as medidas coletadas; • Responder as questões definidas; • Responder ao objetivo definido; • Gerar relatório dos resultados das medições, por meio de indicadores.

Fonte: do autor

Dentre as vantagens da abordagem GQM temos:

- Separa os assuntos organizacionais (metas) dos assuntos específicos de processo produtivo (questões);
- Ajuda na identificação de métricas úteis e relevantes;
- Apóia a análise e interpretação dos dados coletados;
- Permite uma avaliação da validade das conclusões tiradas;
- Focaliza a coleta de dados e sugere que os dados coletados devem ser analisados sob a perspectiva da questão que deve ser respondida;

- Apóia a definição top-down do processo de medição e a análise bottom-up dos dados resultantes;
- Diminui a resistência das pessoas contra processos de medição.

Quadro 6: Abordagem do método GQM

Nível	Aspecto	Questionamento
Conceitual	Goal	Quais são as metas/objetivos?
Operacional	Question	Quais questões se deseja responder?
Quantitativo	Metric	Quais métricas/indicadores poderão ajudar?

Fonte: do autor

5.4 Aprimoramento do Processo

Jones (2000) defende que uma organização deve identificar as maiores áreas de riscos e focalizar as oportunidades de aprimoramento de processos produtivo e gerencial para corrigir ou minimizar os fatores de risco, dando prioridade às melhorias com base em fatores estratégicos de sucesso, críticos para a organização na perspectiva de sustentatibilidade do negócio, como os mencionados a seguir:

- Redução do tempo de entrega para o mercado;
- Redução dos custos de produção e manutenção do software;
- Melhoria na precisão das estimativas de esforço, custos e cronogramas;
- Uso otimizado de novas tecnologias;
- Uso eficiente de consultores e contratos de terceirização.

A coleta de dados históricos de desempenho (cronogramas, custo, qualidade e produtividade) deve ser uma das primeiras ações dentro do processo de melhoria contínua com a finalidade de obter-se uma base de comparação entre projetos e proporcionar entendimento crescente dos processos de software. Utilizam-se os dados do processo do projeto tanto para analisá-lo quanto para modificá-lo no sentido de prevenir problemas e melhorar a eficiência na realização de projetos de software, conforme relata Rocha (2001).

Um método para melhoria contínua no processo é o ciclo PDCA, ciclo de Shewhart ou ciclo de Deming. O ciclo começa pelo planejamento da mudança no processo, em seguida a ação ou conjunto de ações planejadas são executadas, verifica-se o que foi feito para apurar se estava de acordo com o planejado, constantemente e repetidamente (ciclicamente) e, toma-se uma ação para eliminar ou ao menos mitigar defeitos no produto ou na execução.

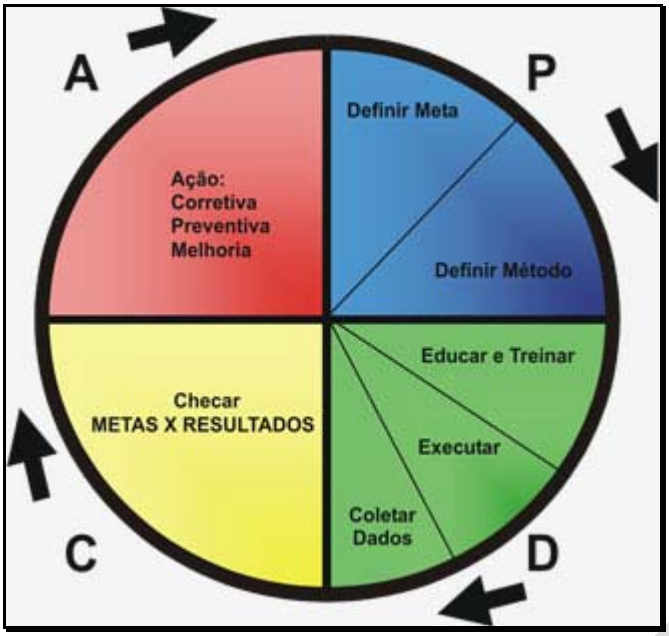


Figura 7: Ciclo PDCA

Fonte: Knower Consultores, 2008

Essencialmente, as fases do ciclo PDCA, enumeradas no Quadro 7, são detalhadas em atividades e produtos que as compõem.

Quadro 7: Detalhamento de fases do ciclo PDCA

Fase	Atividade/Produto
Planejar	• Metas; • Objetivos; • Necessidade de Informações; • Fatores Críticos de Sucesso; • Problemas e Oportunidades.
Fazer	• Criação de medidas, padrões e coleta de dados.
Verificar	• Acompanhar o andamento físico e financeiro dos projetos e serviços; • Verificação dos projetos com desvio de estimativas de prazos; • Verificação dos projetos com desvio de estimativas de custos; • Análise de Impacto (avaliar as variações de qualidade, produtividade e aspectos de custos dos processos de produção e gestão do software, em função de mudanças no ambiente do projeto); • Análise de Tendências (comportamento futuro dos

	principais atributos do processo de planejamento de projetos, produção de software e gestão de produto); • Comparação entre plataformas e tecnologias.
Agir	• Incentivar / Premiar; • Treinar; • Alocar / Realocar Recursos; • Modelar o Ambiente de Desenvolvimento; • Eliminar os problemas antes deles te eliminarem.

Fonte: do autor

5.5 Gerenciamento de Mudanças de Requisitos

Incluir texto técnico.

Tabela 1: Importância da Gerência de Requisitos para Sucesso de Projetos

Fator de Sucesso de Projeto	**Percentual**
Estabelecimento claro dos requisitos	57,2%
Envolvimento do usuário	15,9%
Suporte da Gerência Executiva	13,9%
Outros (marcos do projeto atingíveis, expectativas realistas, planejamento apropriado, gerente de projeto experiente, equipe competente, comprometimento, metodologia formal, gerência de custo, etc)	13,0%

Fonte: Adaptado de CHAOS Reports – The Standish Group, 2004

Incluir texto técnico.

Tabela 2: Importância da Gerência de Mudanças de Requisitos em Projetos

Fator de Estouro de Prazos de Projeto	**Percentual**
Falta de informações do usuário	12,8%
Especificações e requisitos incompletos	12,3%
Mudanças de especificações e requisitos	11,8%
Outros (falta de suporte executivo, incompetência tecnológica, escassez de recursos, objetivos confusos, expectativas irreais, janelas de tempo irreais, etc)	63,1%

Fonte: CHAOS Reports – The Standish Group, 1996

Incluir texto técnico.

Tabela 3: Influência da Gerência de Requisitos em Projetos

Fator de Cancelamento de Projeto	Percentual
Requisitos incompletos	13,1%
Falta de envolvimento do usuário	12,4%
Recursos e prazos insuficientes	10,6%
Mudanças de especificação e requisitos	8,7%
Outros (falta de suporte executivo, expectativas irreais, planejamento pobre, características desnecessárias, falta de gerência de TI, analfabetismo tecnológico, etc)	55,2%

Fonte: Adaptado de CHAOS Reports – The Standish Group, 1996

6 CONSIDERAÇÕES FINAIS

6.1 Críticas

6.2 Sugestões

6.3 Recomendações

REFERÊNCIAS BIBLIOGRÁFICAS

SOMMERVILLE, Ian, Engenharia de Software, 2007, Pearson Education, 8ª. Edição, São Paulo.

PFLEEGER, Shari Lawrence, Engenharia de Software: Teoria e Prática, 2003, Prentice Hall, 2ª. Edição, São Paulo.

ROCHA, Ana Regina Cavalcanti et al., Qualidade de Software: Teoria e Prática, 2001, Prentice Hall, São Paulo.

VAZQUEZ, Carlos Eduardo et al., Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de Software, 2006, Érica, 5ª. Edição, São Paulo.

NBR ISO/IEC 12207:1998 Tecnologia da Informação – Processos de Ciclo de Vida de Software.

MPS.BR – Guia de Implementação – Parte 2 V1.1 – Julho/2007.

MPS.BR – Guia Geral, V1.2 – Associação Para Promoção Da Excelência Do Software Brasileiro – SOFTEX, 2007.

DEMARCO, Tom, Controlling software projects, 1982, Prentice-Hall, New York.

FENTON, N. & PFLEEGER, S.L., Software Metrics: A rigorous and practical approach, 1998, PWS Pub., 2st edition, New York.

BASILI. V.; CALDIERA, G.; ROMBACH, H., Goal Question Metric Paradigm, 1994, Encyclopedia of Software Engineering, v.2, pp: 527 – 532, John Wiley & Sons.

JONES, Capers, Software Assessments, Benchmarks, and Best Practices, 2000, Addison-Wesley, 1st edition, New York.

MEDEIROS, Ernani, Desenvolvendo Software com UML 2.0, 2004, Makron Books, São Paulo.

PRESSMAN, Roger S., Engenharia de Software, 2002, McGraw-Hill, São Paulo.

ANDRADE, E. L. P. Pontos de Caso de Uso e Pontos de Função na Gestão de Estimativa de Tamanho de Projetos de Software Orientados a Objetos, 2004, Dissertação (Mestrado em Gestão do Conhecimento e da Tecnologia da Informação) – Universidade Católica de Brasília, Brasília.

SCHNEIDER, Geri; WINTERS, Jason P., Applying Use Cases: A Practical Guide, 2001, Addison-Wesley, 2st edition, New York.

MAXWELL, Katrina D., Applied Statistics for Software Managers, 2002, Prentice Hall PTR, 1st edition, New York.

EBERT, Christof, et al., Best Practices in Software Measurement, 2004, Springer, 1st edition, New York.

REFERÊNCIAS ELETRÔNICAS

http://www.bfpug.com.br/artigos.htm, Métricas de Software, BFPUG, 29/09/2007.

http://www.fgvam.br/portal/eventos/cicloexcelencia/PMBOK2000.pdf, PMBOK 2000, FGV-AM, 29/09/2007.

http://www.softex.br/mpsbr/, MPS.BR – Melhoria de Processos do Software Brasileiro, SOFTEX, 29/09/2007.

http://www.fattocs.com.br/recursos.asp, Métricas de Software, FATTO, 29/09/2007.

http://www.itmpi.org/, Gerenciamento de Projetos de Software, The IT Metrics and Productivity Institute, 29/09/2007.

http://www.swebok.org, Engenharia de Software, IEEE, 29/09/2007.

http://www.sei.cmu.edu/cmmi/, CMMI – Melhoria de Processos de Software, SEI, 29/09/2007.

http://www.cmmi.de/cmmi_v1.2/browser.html, CMMI – Melhoria de Processos de Software, WIBAS, 29/09/2007.

http://www.standishgroup.com/, Pesquisas de Uso de TI, The Standish Group International, 07/03/2008.

APÊNDICES

Detalhar conteúdo suplementar nesta seção.

ANEXOS

Detalhar conteúdo suplementar nesta seção.