Leila Ribeiro de Oliveira

# IMPLEMENTAÇÃO DE PROCESSOS: O USO DE TÉCNICAS DE ESTIMATIVAS DE PROJETOS DE SOFTWARE PARA ESTIMAR PROCESSOS DE NEGÓCIO

Belo Horizonte
2012

Leila Ribeiro de Oliveira

# IMPLEMENTAÇÃO DE PROCESSOS: O USO DE TÉCNICAS DE ESTIMATIVAS DE PROJETOS DE SOFTWARE PARA ESTIMAR PROCESSOS DE NEGÓCIO

Projeto de dissertação submetido para qualificação no curso de mestrado profissional em sistemas de informação e gestão do conhecimento da Faculdade de Ciências Empresariais da Universidade FUMEC.

Orientador:
Prof. Dr. Fernando Silva Parreiras

Mestrado Profissional em Sistemas de informação e Gestão do Conhecimento
Faculdade de Ciências Empresariais
Universidade FUMEC

Belo Horizonte
2012

**“APRENDER É A ÚNICA COISA DE QUE A MENTE NUNCA SE CANSA, NUNCA TEM MEDO E NUNCA SE ARREPENDE”. LEONARDO DA VINCI**

**SUMÁRIO**

# Lista de Figuras

# Lista de Tabelas

# Resumo

**A**tualmente, as organizações têm voltado sua atenção na busca por melhoria continua de processos que satisfaçam suas necessidades e as tornem mais competitivas frente ao mercado. Várias iniciativas, para identificar e monitorar estratégias e redesenhar processos organizacionais são criadas, porém ainda é percebível a dificuldade que se tem em realizar um controle de tais projetos e principalmente de estimar o custo e o prazo para implementação destes. A prática de gerenciamento de projetos tem-se tornado uma forma de minimizar os impactos causados pela falta de planejamento específico, uma forma de monitorar, acompanhar e organizar os projetos de processos. Porém, como os clientes estão cada vez mais exigentes com a qualidade das entregas, os prazos e principalmente com o custo, observa-se a necessidade de melhor estimar projetos de processos. De forma a contribuir nesta linha de pesquisa, o presente estudo procurará verificar a seguinte questão: "Técnicas de estimativa de processo de software são aplicáveis em projetos de processo?". Para isso o estudo irá abordar algumas técnicas e trabalhos sobre estimativa de esforço já apresentadas na pesquisa do Departamento de Informática Aplicada da UNIRIO e na pesquisa de Kanjan Thammarak entre outros autores renomados, cujos pontos positivos possibilitam a aplicação da estimativa em projetos de processos. O objetivo deste trabalho será investigar e verificar se as técnicas existentes para estimar projetos de software podem contribuir para projetos de desenho e redesenho de processos e propor uma estimativa, baseada nos pontos levantados, voltada para projetos de processos, realizando testes em projetos de redesenho de processos já finalizados e implantados. O presente estudo propõe um levantamento das estimativas que poderiam contribuir para projetos de processos, criação de uma técnica baseada nessas estimativas e a sua aplicação em um ambiente controlado, ou seja, realização de testes em projetos de redesenho de processos já finalizados e implantados. Em seguida, serão comparadas as estimativas realizadas de custo, escopo e prazo reais com a proposta deste projeto. Para melhor compreensão do estudo, este trabalho seguirá uma metodologia fundamentada e utilizada por órgão reconhecido mundialmente - o Practical Software Measurement (PSM) - realizando adaptações quando necessário.

O tipo de pesquisa será descritiva (qualitativa) e os métodos e técnicas utilizados serão: Pesquisa bibliográfica e análise documental; Realização de Experimento; Variável controlada = modelo de estimativa proposto; Variável independente =série histórica com os

dados de projetos passados (previsto e realizado); Comparação das estimativas de projetos passados com os valores resultantes do método proposto.

O experimento será realizado em organizações que já identificaram a necessidade de redesenhar seus processos organizacionais por analistas que já trabalharam pelo menos com uma técnica ou método de mensuração de projetos. Entende-se que para melhor compreensão do estudo, este trabalho deve seguir uma metodologia fundamentada e utilizada por órgãos reconhecidos mundialmente. Logo este estudo seguirá o modelo da o *Practical Software Measurement (*PSM) realizando adaptações quando necessário**.**

Palavras Chaves: Estimativas de Software, Estimativa de Processos, Gerenciamento de Processos, Gerenciamento de Projetos, Métricas.

# CAPITULO 1

# INTRODUÇÃO

Atualmente, as organizações têm voltado sua atenção na busca por melhoria continua de processos que satisfaçam suas necessidades e as tornem mais competitivas frente ao mercado. Várias iniciativas, para identificar e monitorar estratégias, e redesenhar processos organizacionais são criadas, porém ainda é percebível a dificuldade que se tem em realizar um controle de tais projetos e principalmente de estimar o custo e o prazo para implementação destes. A prática de gerenciamento de projetos tem-se tornado uma forma de minimizar os impactos causados pela falta de planejamento específico, uma forma de monitorar, acompanhar e organizar os projetos de processos. Porém, como os clientes estão cada vez mais exigentes com a qualidade das entregas, os prazos e principalmente com o custo, observa-se a necessidade de melhor estimar projetos de processos.

O Departamento de Informática Aplicada da UNIRIO/NP2TEC, em 2009, realizou uma busca sistemática na literatura da Ciência da Computação, em particular na área de Engenharia de Software, por trabalhos sobre estimativas de esforço para a realização de projetos de modelagem de processos de negócio e desenvolvimento de software que possam elucidar mecanismos aplicáveis aos projetos de modelagem de processos de negócio na Petrobrás. O estudo levou à equipe a conclusão que a realização da pesquisa foi acertada, uma vez que apesar da existência de artigos sobre métricas de qualidade para modelos de processos de negócio, não foi encontrada nenhuma publicação que abordasse ou fizesse referência a algum método ou técnica para realizar estimavas em projetos de modelagem de processos. De acordo com o estudo, alguns autores ressaltam a similaridade entre projetos de desenvolvimento de software e projetos de modelagem de processos de negócio e, baseados nesta similaridade, propõe uma customização das métricas de software para área de modelagem de processos de negócio. Outro estudo encontrado foi de Kanjan Thammarak do departamento de informática da faculdade de administração de Bangkok na Tailândia. O objetivo principal do estudo é examinar algumas métricas conhecidas e verificar os ganhos que tais métricas podem trazer para a sua utilização e reutilização em processos de negócios de forma qualitativa e quantitativamente.

De forma a contribuir nesta linha de pesquisa, o presente estudo procurará verificar a seguinte questão: “Técnicas de estimativa de processo de software são aplicáveis em projetos de processo?”. Para isso o estudo irá abordar algumas técnicas e trabalhos sobre estimativa de esforço já apresentadas na pesquisa do Departamento de Informática Aplicada da UNIRIO e na pesquisa de Kanjan Thammarak entre outros autores renomados, cujos pontos positivos possibilitam a aplicação da estimativa em projetos de processos. O presente estudo propõe um levantamento das estimativas que poderiam mais contribuir para projetos de processos, criação de uma técnica baseada nessas estimativas e a sua aplicação em um ambiente controlado, ou seja, realização de testes em projetos de redesenho de processos já finalizados e implantados. Em seguida, serão comparadas as estimativas realizadas de custo, escopo e prazo reais com a proposta deste projeto. Para melhor compreensão do estudo, este trabalho seguirá uma metodologia fundamentada e utilizada por órgão reconhecido mundialmente - o *Practical Software Measurement* (PSM) - realizando adaptações quando necessário.

### 1.1. Justificativa

Atualmente, as organizações têm voltado sua atenção na busca por melhoria continua de processos que satisfaçam suas necessidades e as tornem mais competitivas frente ao mercado. Várias iniciativas, para identificar e monitorar estratégias, e redesenhar processos organizacionais são criadas, porém ainda é percebível a dificuldade que se tem em estimar o custo e o prazo para implementação de tais projetos.

Em geral, um projeto de processos parte de um novo escopo e contexto e espera-se uma solução única dentro do prazo e custo programado que satisfaça a necessidade do cliente. Logo, de maneira geral, práticas de gerência de projetos são utilizadas também para estes tipos de projetos e estimativas de esforço precisam ser especificadas tanto no inicio quanto na fase de análise de viabilidade do projeto. Sabe-se que problemas com prazos, custo e escopo em projetos, principalmente T.I, constituem um fator crítico de insucesso e, normalmente isso ocorre por problemas nas estimativas iniciais dos projetos. Os prazos e o custo devem ser pelo menos proporcionais ao escopo definido para o projeto e consequentemente ao tamanho da equipe e recursos disponíveis. Projetos de processos estão sujeitos aos mesmos problemas e

riscos. Rovai, Silva e Campanário (2004), no primeiro congresso internacional de Gestão de Tecnologia e Sistemas de Informação (CONTECSI) na USP/São Paulo, cita Goldberg et al, (1998) ao apontar a criticidade das estimativas em projetos de TI. De acordo com esses autores, Goldberg (1998) destaca em seu trabalho alguns riscos e restrições de projetos de TI, conforme abaixo:

- Complexidade e indefinições de escopo;
- Criticidade de prazos;
- Orçamentos com estimativas irrealistas ou não devidamente parametrizadas;
- Perda ou inversão de prioridade;
- Descontrole de escopo, ou falta de um sistema de controle de configuração e alterações de escopo;
- Conflitos entre prioridades e disputa entre recursos de projetos concorrentes;
- Problemas relativos à determinação da produtividade dos desenvolvedores e sua respectiva qualificação e graduação;
- E principalmente a não utilização de metodologias estruturadas para o processo de estimativas do uso de recursos.

Tais problemas apresentados, também aparecem em projetos de processos de negócio. Assim, utilizar métodos e técnicas eficazes para estimar de maneira mais precisa os projetos de processos afetará de forma significativa os resultados esperados em cada entrega e sabe-se que atualmente várias técnicas existem para estimativas de esforço, porém para projetos de desenvolvimento de software. A questão a ser investigada nesse trabalho será “Técnicas de estimativa de processo de software são aplicáveis em projetos de processo?”.

O objetivo deste trabalho será investigar e verificar se as técnicas existentes para estimar projetos de software podem contribuir para projetos de desenho e redesenho de processos e propor um método para estimar (nova estimativa), baseado nos pontos levantados, voltada para projetos de processos, realizando testes em projetos de redesenho de processos já finalizados e implantados.

## 1.2. Objetivos: Geral e Específico

O objetivo geral desse trabalho é analisar a aplicação das técnicas existentes para estimar projetos de software em projetos de desenho e redesenho de processos e propor um método para estimar (estimativa), baseado nos pontos positivos levantados, voltado para projetos de processos. Para tanto, este trabalho se propôs aos seguintes objetivos específicos:

a) Analisar técnicas e estimativas de processo de software;

b) Desenvolver um método para estimar esforço para projetos de redesenho de processos;

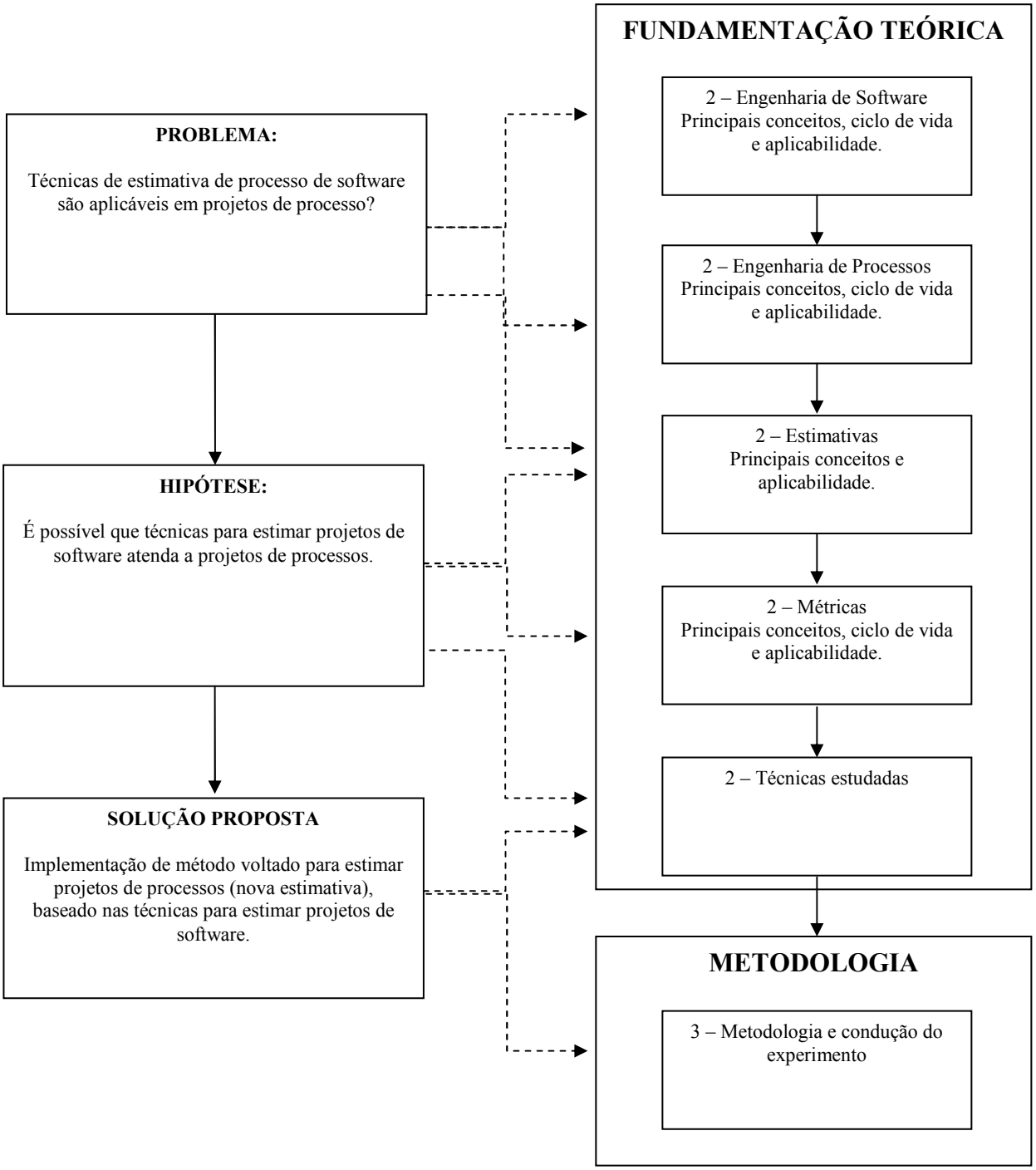
c) Verificar e validar a aplicabilidade do modelo proposto;

d) Analisar se os valores estimados para os projetos correspondem aos valores reais (esforço realizado).

## 1.3. Organização do Projeto

Esse trabalho foi organizado em três capítulos. Na introdução são apresentados além da justificativa do estudo, o objetivo geral do estudo e os objetivos específicos. O referencial teórico apresenta conceitos de processos de desenvolvimento de software e, conceitos sobre engenharia de processos, além de apresentar conceitos fundamentais para o estudo como “estimativas”, “métricas, e técnicas de estimativas, cujo objetivo é a verificação se sua utilização ou se parte dela, atende a projetos de processos. No capitulo 3 tanto a metodologia da pesquisa é apresentada quanto à forma que será conduzida o experimento, além do cronograma deste estudo.

A Figura 1 apresenta graficamente a organização do pré-projeto em capítulos.



# CAPITULO 2
# FUNDAMENTAÇÃO TEÓRICA

## 2.1. Engenharia de Software

As organizações têm voltado sua atenção na busca por melhoria continua de processos que satisfaçam suas necessidades e as tornem mais competitivas frente ao mercado. O desenvolvimento de software nesse sentido torna-se uma atividade de grande importância nessa corrida para o sucesso. A adoção de metodologias para padronização e gerência nos processos de desenvolvimento de software com o intuito de buscar qualidade nos processos e produtos de software proporciona ganho de competitividade frente ao mercado, pois garante serviços ou produtos de forma rápida, com custos mais baixos e com mais qualidade e segurança. De acordo com Maffeo (1992), a engenharia de software tem como objetivo primário o aprimoramento da qualidade dos produtos de software e o aumento da produtividade dos engenheiros de software, além do atendimento aos requisitos de eficácia e eficiência, ou seja, efetividade. Logo, a engenharia de software veio para auxiliar as organizações a produzirem com rapidez, com maior qualidade e custo mais baixos, garantindo assim, um ganho frente ao mercado.

Através da engenharia de software é possível padronizar a forma de metrificar e gerar estimativas, além de auxiliar na definição de quando medir e quando estimar, aprimorando assim a qualidade nas entregas, no custo e nos prazos estabelecidos. Segundo Sommerville (2011), a engenharia de software é uma disciplina da engenharia que se preocupa com todos os aspectos da produção de software desde o início da especificação do sistema até a manutenção do sistema após esse estar sendo usado. De acordo com Pressman (2002), engenharia de software é a união de três elementos fundamentais: métodos, ferramentas e procedimentos. Os métodos especificam como desenvolver o software, as ferramentas possibilitam o meio para construção do software de forma automatizado ou semi-automatizado apoiando o método; e os procedimentos constituem o elo entre os métodos e as ferramentas. Da mesma forma que desenvolver um software, a forma de estimar projetos, também necessita de um método, uma ferramenta e um procedimento e um auxilia o outro, pois se um produto é bem medido sua qualidade conseqüentemente tende a aumentar. Segundo Pressman (1995, p. 56):

> Na maioria dos empreendimentos técnicos, as medições e as métricas ajudam-nos a entender o processo técnico usado para se desenvolver um produto, como também o próprio produto. O processo é medido, num esforço para melhorá-lo. O produto é medido, num esforço para aumentar sua qualidade.

### 2.1.1. Processo de Software

Durante a crise do Software na década de 70 os produtos oriundos de desenvolvimento de software eram de má qualidade e isso era devido à forma desestruturada e sem planejamento que as atividades para desenvolvimento de software eram executadas (Melo, 2004). A partir deste cenário, surgiu a necessidade de tornar o desenvolvimento de software um processo estruturado, planejado e padronizado.

Atualmente existe uma grande ênfase na análise de processos nas organizações. Segundo Chiavenato (2000, p.49) processo pode-se caracterizar como a transformação de entrada (requisitos) em produtos acabados (saída) podendo ter retorno de capital investido, ou seja, lucro. Já Pressman (2006, p.17) descreve:

> Os processos de software formam a base para o controle gerencial de projetos de software e estabelecem o contexto no qual os métodos técnicos são aplicados, os produtos de trabalho (modelos, documentos, dados, relatórios, formulários etc.) são produzidos, os marcos são estabelecidos, a qualidade é assegurada e as modificações são adequadamente geridas.

Pressman (2006), ainda explica que a engenharia de software é uma tecnologia em camadas que deve se apoiar num compromisso organizacional com qualidade. De acordo com o autor, o alicerce que sustenta essa tecnologia da engenharia de software é a camada de processo - Pressman (2006, p.17):

> O processo de engenharia de software é o adesivo que mantém unidas as camadas de tecnologia e permite o desenvolvimento racional e oportuno de softwares de computador. O processo define um arcabouço [PAU93] que deve ser estabelecido para a efetiva utilização da tecnologia de engenharia de software.

Figura 1 ilustra a camada de processo e as demais camadas da engenharia de software.


**Figura 1 - Camadas da Engenharia de Software (Fonte: Pressman, 2006)**

Os processos de software, de acordo com Pressman (2006), formam a base para o controle gerencial de projetos de software e estabelece o contexto no qual os métodos técnicos são aplicados, os produtos de trabalho são produzidos, os marcos são estabelecidos, a qualidade é assegurada e as modificações são adequadamente geridas. Já os métodos fornecem a técnica de "como fazer" para construir softwares. Os métodos abrangem um amplo conjunto de tarefas que incluem comunicação, análise de requisitos, modelagem de projeto, construção de programas, testes e manutenção. E as ferramentas fornecem apoio automatizado ou semi-automatizado para o processo e para os métodos. Para Sommerville (2007), processo de software é um conjunto estruturado de atividades necessárias para o desenvolvimento de um sistema de software. O autor observa que embora existam muitos processos de software diferentes, todos possuem algumas atividades em comum, como:

- Especificação: onde os clientes e engenheiros definem o software que deve ser produzido e as restrições sobre o seu funcionamento, ou seja, nessa atividade, a funcionalidade do software e as restrições sobre sua operação devem ser definidas;
- Projeto e implementação: o software é projetado e programado;
- Validação: o software deve ser validado para garantir de atender ao que o cliente necessita;
- Evolução: O software é modificado para refletir as mudanças de requisitos do cliente e do mercado, ou seja, o software deve evoluir para em resposta a mudanças nas necessidades do cliente.

Pressman (2006, p.19), também entende que independente do modelo de processo utilizado, um arcabouço de processos genéricos é aplicável a grande maioria dos processos de software, independente de seu tamanho ou complexidade. Da mesma forma Sommerville (2007), Pressman enfatiza que o arcabouço de processos engloba um conjunto de atividades guarda-chuva que são aplicáveis durante todo o processo de software e a forma de organizar e utilizar as atividades, ou seja, em sequência ou intercaladas, podem variar de acordo com o

modelo de desenvolvimento de software utilizado e independente do modelo processo de software utilizado, algumas atividades podem ser utilizadas de forma genérica:

- Comunicação: envolve alta comunicação e colaboração com o cliente e outros interessados e abrange o levantamento de requisitos e outras atividades relacionadas.
- Planejamento: estabelece um plano para o trabalho de desenvolvimento do software, ou seja, descrevem as tarefas técnicas a serem conduzidas, os riscos do projeto, os recursos necessários, os artefatos a serem produzidos e um cronograma de trabalho;
- Modelagem: cria os modelos que permitem entender melhor os requisitos do software e o projeto que vai satisfazer esses requisitos. É a representação de engenharia do software que vai ser construído;
- Construção: combina a geração do código-fonte, que deve implementar as funcionalidades especificadas, e os testes necessários para descobrir os erros na função, no comportamento e no desempenho do software;
- Implantação: entrega do software ao cliente, que avalia o produto e fornece feedback com base na avaliação.

Sendo assim, pode-se dizer que para se construir um produto de software, independente do modelo de software utilizado, é necessário seguir modelos padronizados, ou seja, atividades pré-estabelecidas para transformar requisitos em produtos finais, dentro de um escopo de trabalho pré-definido com o objetivo de atingir um resultado com qualidade e em um tempo previsto.

### 2.1.2. Modelos de Desenvolvimento de Software

Um modelo de processo de software (ciclo de vida), é uma descrição simplificada de um processo de software, é uma representação abstrata de um processo para explicar as diferentes abordagens de desenvolvimento (Sommerville, 2003).

De acordo com o PMBOK (2008), um ciclo de vida pode ser documentado como uma metodologia e consiste nas fases do projeto que geralmente são sequenciais

e que às vezes se sobrepõem, cujo nome e número são determinados pelas necessidades de gerenciamento, controle das organizações envolvidas, natureza do projeto entre outras características.

Vários modelos de desenvolvimento de software, propostos pela literatura, são utilizados atualmente. Os tópicos a seguir, fundamentados em Pressman (2006), Somerville (2007), apresentam os modelos mais utilizados e considerados importantes para entendimento nessa dissertação.

#### 2.1.2.1. Modelo Cascata

O modelo cascata, também chamado de ciclo de vida clássico, foi proposto por Royce em 1970. Até meados da década de 1980 foi o único modelo com aceitação geral. De acordo com Pressman (2006), no modelo cascata as fases definidas para o desenvolvimento do software são sistematicamente seguidas de maneira seqüencial. O modelo inicia com a fase de especificação dos requisitos pelo cliente e progride ao longo do planejamento, modelagem, construção e implantação, culminando na manutenção progressiva do software acabado como apresentado na Figura 2.

Segundo Pressman (2006, p.39), o modelo em cascata é o paradigma mais antigo, no entanto, já em 2006, esse autor relata que nas últimas décadas, sua eficácia é questionável devido alguns problemas levantados quando o modelo é aplicado. Os principais problemas, de acordo com o autor, se devem ao fato que projetos reais raramente seguem o fluxo seqüencial que o modelo propõe; este modelo exige estabelecer todos os requisitos na fase inicial, fato que em geral é difícil tanto para o cliente quanto para o desenvolvedor, já que os requisitos tendem a mudar constantemente. Outro problema é a demora na entrega de uma versão executável do software, a versão executável somente fica disponível para o cliente no período final do intervalo de tempo do projeto, ou seja, no final do projeto.

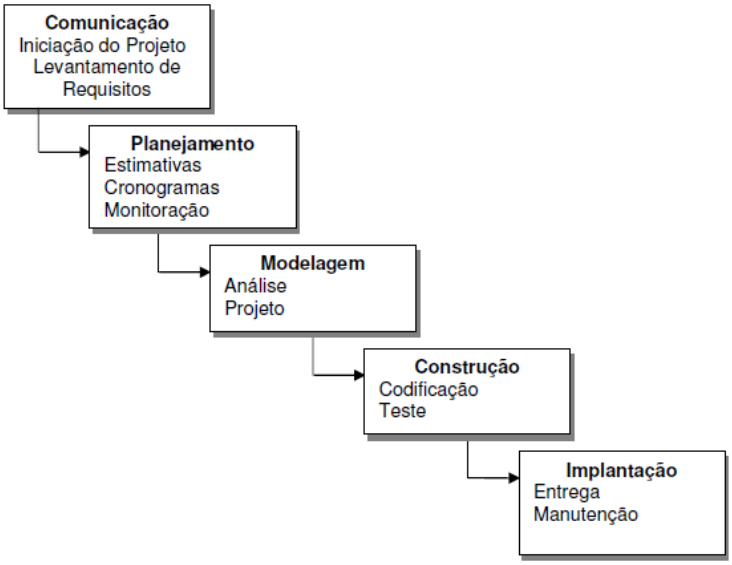



**Figura 2 - Modelo Cascata (fonte: Pressman, 2006)**

### 2.1.2.2. Modelos Incrementais

O modelo incremental entrega uma série de versões chamadas de incrementos, que fornecem progressivamente mais funcionalidades para os clientes à medida que cada incremento é entregue. De acordo com Pressman (2006, p.40), o modelo incremental combina elementos do modelo cascata aplicado de maneira iterativa. O primeiro incremento frequentemente é chamado de núcleo do produto e nele estão contidos os requisitos básicos do sistema. O núcleo do produto é usado pelo cliente ou passa por uma revisão detalhada e um plano é desenvolvido para o próximo incremento como resultado do uso e/ou avaliação. O plano visa o aperfeiçoamento do núcleo do produto para melhor satisfazer às necessidades do cliente. Os próximos incrementos desenvolvidos agregarão os requisitos do núcleo do produto e dos incrementos anteriores. O modelo incremental tem o objetivo de apresentar um produto operacional a cada incremento, e é particularmente útil quando não há mão-de-obra e/ou recursos técnicos disponíveis para a implementação completa, dentro do prazo comercial de entrega estabelecido para o projeto. Este modelo é um melhoramento do modelo em cascata, pois permite a alteração dos requisitos durante o desenvolvimento do software.

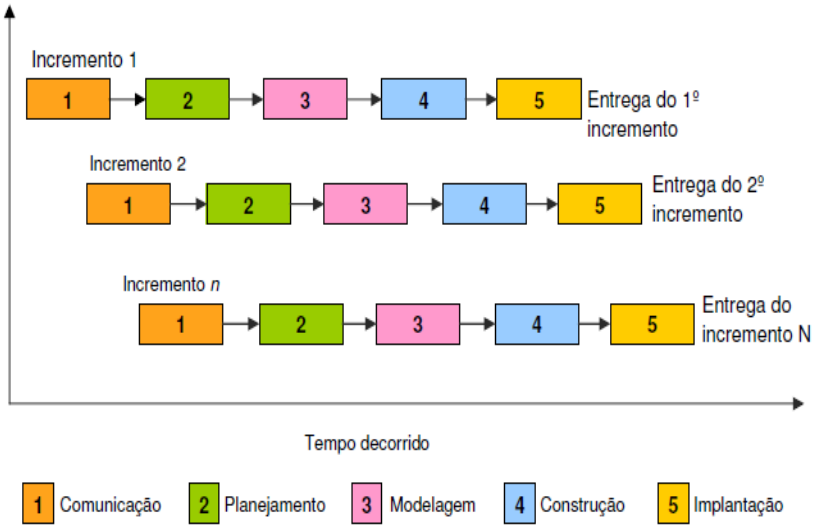



**Figura 3 - Modelo Incremental (fonte: Pressman, 2006)**

### 2.1.2.3. Modelos Evolucionários

Modelos evolucionários de processo produzem uma versão cada vez mais completa do software a cada iteração. De acordo com Pressman (2006), os modelos evolucionários de processo reconhecem a natureza iterativa da maioria dos projetos de engenharia de software e são projetados para acomodar as modificações necessárias, expondo o resultado ao cliente e refinando esse resultado por meio de várias versões, para que então seja desenvolvido o sistema capaz de atender as necessidades do cliente. Esses modelos podem ser adotados para serem aplicados ao longo de todas as atividades de engenharia de software. São exemplos de Modelos Evolucionários: o modelo de prototipagem e o modelo espiral.

#### 2.1.2.3.1. Prototipagem

A prototipação é uma técnica para ajudar engenheiros de software e clientes a entender o que está sendo construído quando os requisitos não estão claros. Com este

modelo uma prévia avaliação do cliente e dos engenheiros pode ser feita. De acordo com Pressman (2006), apesar de a prototipagem poder ser utilizada como um modelo de processo independente, ela é mais comumente utilizada como uma técnica que pode ser implementada dentro do contexto de qualquer um dos modelos de processos existentes. A Figura 4 apresenta o ciclo de desenvolvimento do modelo de prototipagem.

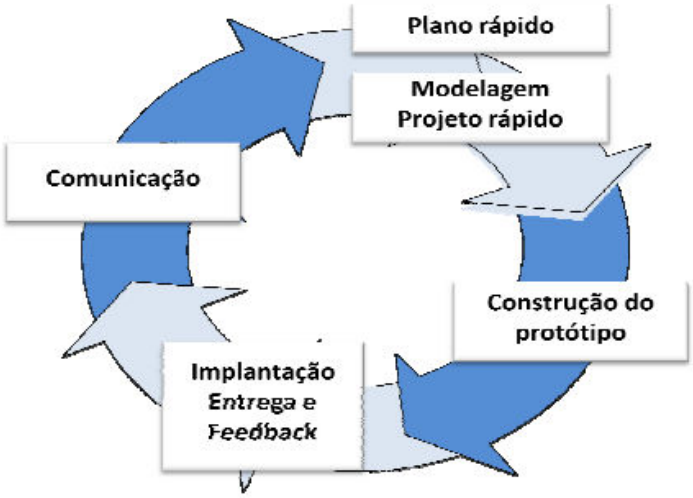



**Figura 4 - Modelo de Prototipagem (fonte: Pressman, 2006)**

De acordo com Sommerville (2003), um protótipo é uma versão inicial de um sistema usada para demonstrar conceitos e testar opções de projeto podendo ser utilizado no processo de engenharia de requisitos para ajudar na elicitação e validação de requisitos, nos processos de projeto para explorar opções e desenvolver um projeto de interface de usuário e no processo de testes para executar testes fim-a-fim. A Figura 5 exemplifica o processo de desenvolvimento de protótipos.

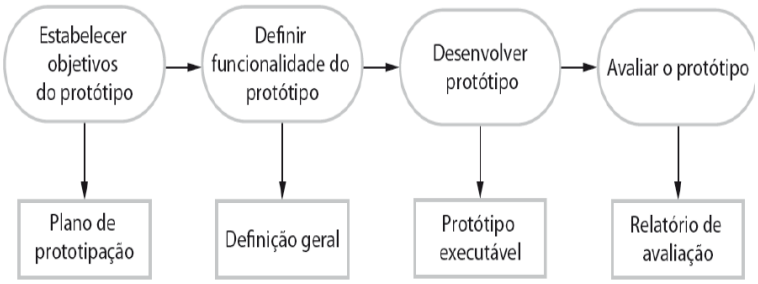



**Figura 5 - Processo de desenvolvimento de Protótipos (fonte: Sommerville, 2003)**

#### 2.1.2.3.2. Espiral

O modelo espiral de acordo com Pressam (2006), originalmente foi proposto por Boenh, é um modelo evolucionário que combina a natureza iterativa de prototipagem com os aspectos controlados e sistemáticos do modelo cascata. Este modelo fornece potencial para desenvolvimento rápido de versões de software cada vez mais completas. De acordo com Sommerville (2003), este processo é representado como uma espiral, ao invés de uma seqüência de atividades com retorno, onde cada volta na espiral representa uma fase no processo. Não existem fases fixas como especificação ou projeto, de maneira que as voltas na espiral são escolhidas de acordo com o que é requerido e os riscos são explicitamente contados e resolvidos durante todo o processo. Dessa forma, este modelo diferentemente de outros modelos, acompanha toda a vida do software, mesmo depois da entrega ao cliente e as falhas do software são identificadas e corrigidas, impedindo que se propaguem para as próximas iterações do ciclo de desenvolvimento do software.

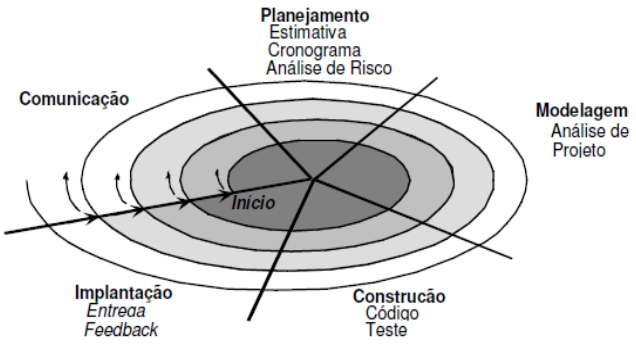



**Figura 6 - Modelo Espiral (fonte: Pressman, 2006)**

### 2.1.2.4. Desenvolvimento baseado na reutilização

Este processo baseia-se na reutilização e na literatura o encontramos também como modelo baseado em componentes. De acordo com Sommerville (2003), o processo se baseia no reuso sistemático em que os sistemas são integrados com componentes existentes ou sistemas COTS (C*ommercial-off-the-shelf*). Para Pressman (2006), o modelo de desenvolvimento baseado em componentes leva ao reuso do

software, e a reusabilidade fornece aos engenheiros vários benefícios mensuráveis. A Figura 7 apresenta as fases do processo baseado em reutilização.

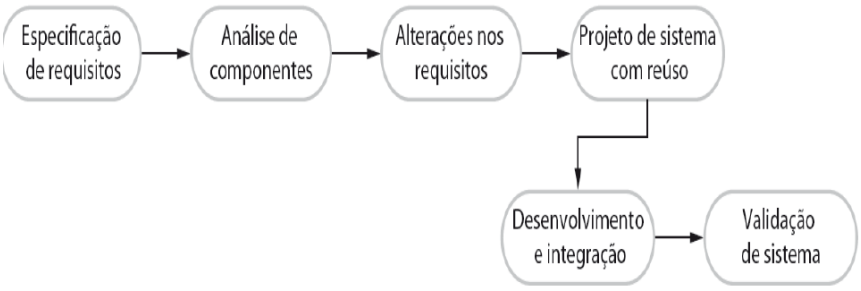



**Figura 7 - Fases do Processo Orientado ao Reuso (fonte: Sommerville, 2003)**

### 2.1.3. Estimativas

Para planejar com eficiência os projetos de desenvolvimento de software, as empresas precisam estimar esforço e custo que serão empreendidos em seus projetos. Estimativas fornecem os dados que permitem prever a quantidade de pessoas que serão necessárias, o tempo necessário e os custos do projeto. De acordo com Pressman (2006), estimativa pode ser considerada como a tentativa de determinar quanto dinheiro, esforço, recursos e tempo serão necessários para construir um sistema ou produto baseado em software específico.

Estimativas de projetos de software apóiam a gerência de projetos e devem ser atividades definidas dentro do processo de desenvolvimento de software. Para os gestores de projetos, estimar não é uma atividade fácil, pois entender e realizar previsões quantitativas, quando somente informações qualitativas são fornecidas ou são as únicas disponíveis, envolve, além de muita audácia, quando confiança em sua própria experiência nas práticas de gestão. Contudo, a forma de estimar, não precisa ser realizada de forma aleatória, pois técnicas disponíveis na literatura podem ser utilizadas. Segundo Pressman (2006, p. 520):

> Apesar de estimar ser tanto arte como ciência, essa importante atividade não precisa ser conduzida de modo aleatório. Já existem técnicas úteis para estimativa de tempo e esforço. Métricas de processos e projetos podem fornecer perspectiva histórica e insumo poderoso para a geração de estimativas quantitativas. Experiência anterior (de todo o pessoal envolvido) pode ajudar imensamente, à medida que estimativas são desenvolvidas e revistas. Como a estimativa estabelece uma base para todas as

> outras atividades de planejamento de projeto e como este fornece um guia ara a engenharia de software bem sucedida, seria temerário começar sem ela.

Estimativas, quando bem elaboradas, tornam-se a base para um bom planejamento de projeto de software, e esta atividade deve ser considerada como o primeiro passo da fase de planejamento. Um planejamento eficiente e um controle efetivo em projetos de software, tais como, planejar cronogramas, orçamentos, respostas a possíveis riscos, sem a utilização de estimativas confiáveis, não seria possível. Estimativas eficientes permitem a verificação da viabilidade dos projetos e confecção de propostas técnicas mais corretas. De acordo com Vazquez (2003), o processo de estimativa de um projeto de software envolve quatro atividades básicas: estimar o tamanho do produto a ser desenvolvido; estimar o esforço empregado na execução do projeto; estimar o prazo (duração) do projeto; estimar o custo do projeto. A forma adotada pelas organizações para estimar projetos de software pode variar de acordo com o modelo de processo utilizado em cada uma e do tipo de projeto, porém é desejável que haja um conhecimento prévio sobre técnicas de estimativas, além de uma visão, mesmo que macro, do escopo do projeto e, se possível, uma base histórica onde seja possível consultar estimativas de projetos semelhantes já finalizados. De acordo com Sommerville (2003), normalmente as seguintes técnicas são utilizadas para a realização de estimativas:

- Técnicas baseadas em experiências – As estimativas de requisitos de futuros esforços são baseadas na experiência do gerente com projetos anteriores e no domínio da aplicação. Essencialmente, o gerente faz um juízo fundamentado a respeito de como devem ser os requisitos de esforço.
- Modelagem algorítmica de custos – Nessa abordagem, é usada uma abordagem para calcular o esforço do projeto com base em estimativas dos atributos de produto, como tamanho e características do processo, assim como a experiência da equipe envolvida.

Já Pressman (2006), propõe alguns passos sistemáticos, que no seu ponto de vista, fornecem estimativas com riscos aceitáveis e confiáveis, conforme abaixo:

- Adiar a estimativa até que o projeto esteja mais adiantado. Essa opção pode ser atrativa, mas no normalmente não é prática. Cada vez mais, em projetos, existe a necessidade de estimar o quanto antes para que seja possível entender o "quanto" deverá ser empreendido e desta forma verificar se o proposto é viável ou não para a organização para aquele momento.
- Basear-se em estimativas de projetos semelhantes, que já foram completados. Neste caso, tanto o projeto, quanto as variáveis que influenciam o projeto (cliente, condições do negócio, etc.) precisam ser semelhantes.
- Utilizar técnicas de decomposição relativamente simples para gerar estimativas de custo e esforço do projeto.
- Utilizar um ou mais modelos empíricos para estimativas de custo e esforço do software.

As estimativas de projetos de software apóiam a gerência de projetos e é recomendável que sejam as primeiras atividades definidas dentro do processo de desenvolvimento de software. Normalmente as estimativas fazem parte das primeiras atividades do planejamento do projeto e é nesta atividade que ocorre a primeira aplicação das métricas. Após o refinamento de requisitos, normalmente, é realizada uma nova estimativa para entendimento da viabilidade do projeto e na finalização do projeto, para entendimento do quanto a estimativa foi acertiva ou não, e assim ter dados históricos para futuros projetos e realização de medições confiáveis. De acordo com Pressman (2006, p. 502):

> A primeira aplicação das métricas de processo de projeto, na maioria dos projetos de software, ocorre durante a estimativa. Métricas coletadas de projetos anteriores são usadas como base, a partir da qual estimativas de esforço e de tempo são feitas para o trabalho atual de software. Conforme o projeto prossegue, medidas de esforço e de tempo despendidos são comparadas com as estimativas originais (e do cronograma proposto). O gerente de projeto usa esses dados para monitorar e controlar o progresso.

Dessa forma, é entendido que estimativas estão fortemente ligadas ao processo de medição, onde são verificados vários indicadores de desempenho de forma quantitativa. De acordo com Pressman (2006), medição pode ser utilizada ao longo de um projeto de software para auxiliar na estimativa, no controle de qualidade, na avaliação de produtividade e no controle do projeto.

### 2.1.4. Métricas

Métricas são medidas quantitativas que permitem aos engenheiros de software ter idéia da eficácia do processo de software e dos projetos que são conduzidos usando o processo como arcabouço (PRESSMAN, 2006). De acordo com o autor, métricas também são utilizadas para detectar grupos de problema, de modo que soluções possam ser desenvolvidas, e que o processo de software possa ser melhorado. De acordo com Pressman (2006, p. 500):

> Medição é uma ferramenta de gestão. Se conduzida adequadamente, fornece conhecimento a um gerente de projetos. E, como resultado, apóia o gerente de projeto e a equipe de software na tomada de decisão que irão conduzir a um projeto de sucesso.

De acordo com Sommerville (2003), pode-se dizer que métricas é qualquer tipo de medição em um sistema, processo ou documentação. As métricas permitem que o software e o processo de software sejam quantificados, elas podem ser utilizadas para prever os atributos de produto ou para controlar o processo de software. Além disso, as métricas de produto podem ser usadas para previsões gerais ou para identificar os componentes anômalos, por exemplo, medidas de tamanho de um produto, número de defeitos relatados em um produto entregue, número de pessoas-dia para desenvolver um componente de um sistema.

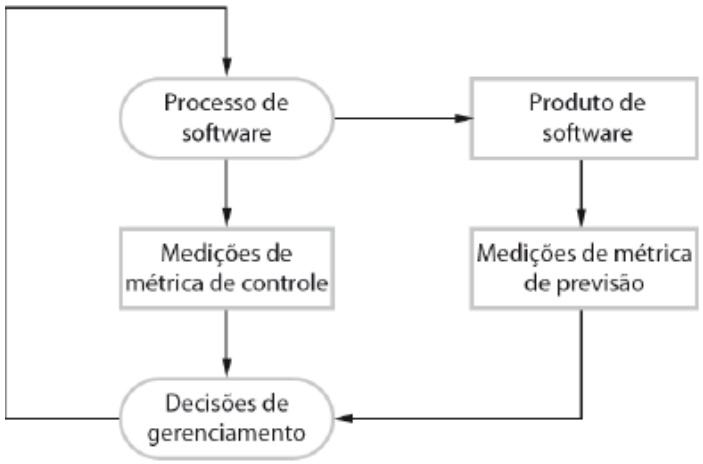



**Figura 8 - Medições de Previsão e Controle (fonte: Sommerville, 2003)**

Existe uma interseção entre estimar e metrificar processo e produtos de software, uma vez que, em processos de desenvolvimento de software é necessário estimar, planejar, medir para posteriormente comparar com o realizado e assim, possibilitar melhores tomadas de decisão, melhor plano de resposta a riscos e constante refinamento das estimativas para os projetos subseqüentes. Para a realização de métricas com qualidade é importante estabelecer passos que auxiliem a coletar, calcular e analisar as métricas para obtenção de melhores resultados. A Figura 9 ilustra um exemplo de sequência de passos para a realização de medição.

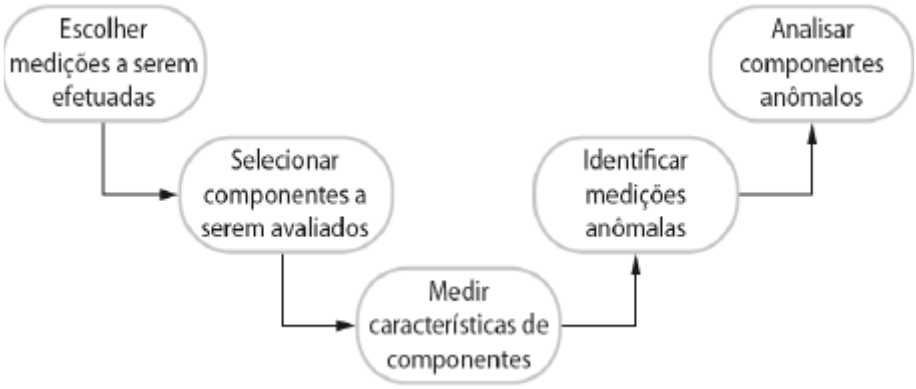



**Figura 9 - Processo de medição (fonte: Sommerville, 2003)**

De acordo com Pressman (2006), o objetivo de metrificar projetos é duplo, uma vez que elas podem ser utilizadas para minimizar o cronograma de desenvolvimento, fazendo os ajustes necessários para evitar atrasos, problemas, e riscos em potencial; e são utilizadas também para avaliar a qualidade do produto durante sua evolução e, quando necessário, modificar a abordagem técnica para aperfeiçoar a qualidade. Logo, as métricas é o resultado direto do aumento da qualidade e consequente diminuição de defeitos e retrabalho no projeto, auxiliando assim na redução do custo total dos projetos.

## 2.2. Engenharia de Processos

### 2.2.1. Reengenharia

A busca por serviços e produtos acessíveis, de maior qualidade e com menor custo aumenta a necessidade das organizações aperfeiçoarem e inovarem seus processos de

negócio. Essa busca está ligada diretamente a inovação e aperfeiçoamento em processos que ocorre desde a revolução industrial. Taylor foi o primeiro a estudar a organização do processo do trabalho manual desenvolvendo e estudando as especialidades do processo industrial - estudo de tempos em 1881, quando Taylor o introduziu na usina da *Midvale Steel Company*. Este estudo contribuiu na determinação de tempos padrão para as operações de processo. Outro estudo de grande importância, desenvolvido na mesma época, para melhoria de métodos de trabalho foi o estudo de movimentos, desenvolvido pelo casal Gilbreth. De acordo com Barnners (1977), em 1930 se iniciou um movimento geral para estudar "o trabalho" com o objetivo de descobrir métodos melhores e mais simples de se executar uma tarefa. Logo, desde então, vários estudos surgiram, um conjunto de métodos, abordagens sobre processos e a melhor forma de melhorá-los foram desenvolvidas e estudadas até os dias atuais. Entre os métodos e técnicas, os mais conhecidos, de acordo com Barros (2009) são:

- Gestão do conhecimento (NONAKA e TAKEUCHI, 1995; CARDOSO, 2004);
- Balance Scorecard - BSC (KAPLAN e NORTON, 1997);
- Cadeia de Valor de Porter (PORTER, 1985);
- Controle da Qualidade Total (CAMPOS, 1996; DEMING, 1990);
- Teoria das Restrições (GOLDRATT, 2003);
- *Workflow* (VALADARES, 2001; VAN DER ALST et. al, 2003);
- Gestão de processos (JESTON e NELIS, 2008a, HARMON, 2005; ROSEMANN, 2008; HAMMER, 2007; PAIM, 2007; PAIM et. al, 2007a; 2008, 2009a);
- Seis Sigma (PYZDEK, 2003);
- Redesenho de processos (SCHEER, 1998);
- Modelos de maturidade de processo (HAMMER, 2007; ROSEMANN e BRUIN, 2005; HARMON, 2005);
- Metodologia BPMN (OMG, 2007).

Barros (2009), em seu estudo, cita essas técnicas e métodos agrupados em vertentes de acordo com HARMON (2005, apud Barros, 2009): a primeira, qualidade, onde o foco é a melhoria da eficiência dos processos; a segunda, gestão, onde o foco é o desempenho organizacional e a terceira, tecnologia da informação, onde o foco é a automação dos processos. A Figura 10 (HARMON, 2005; PAIM, 2008; JESTON e NELIS, 2008 a apud Barros, 2009) apresenta um resumo das técnicas e métodos utilizadas

ao longo das últimas décadas, para integração das diversas iniciativas de processos, até os dias atuais.

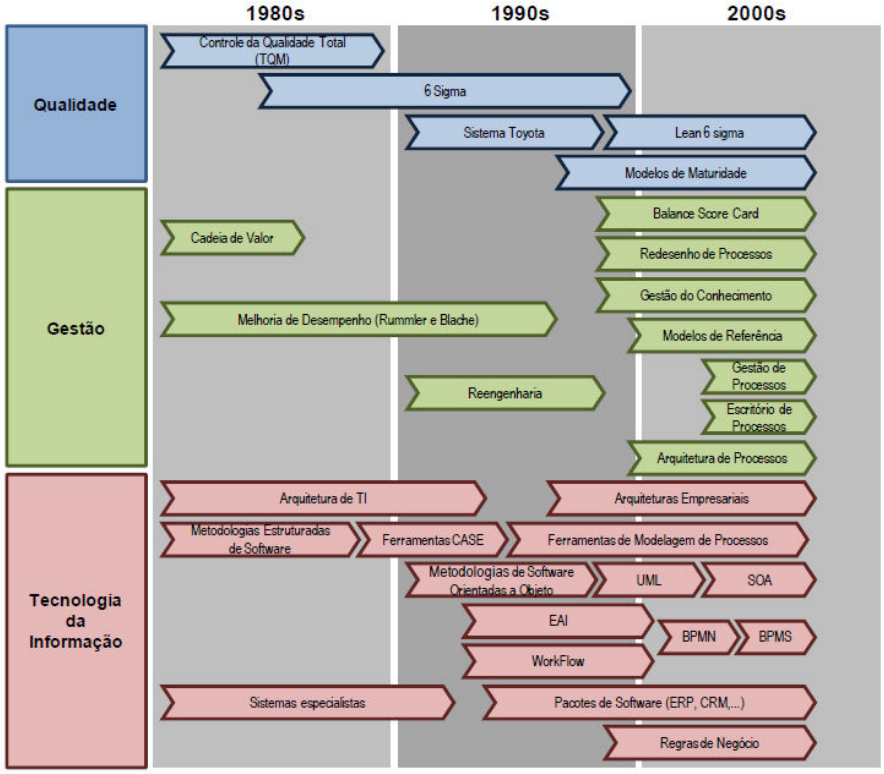



**Figura 10 - Resumo histórico dos métodos e abordagens em processos (HARMON, 2005; PAIN, 2008; JESTON e nelis, 2008 a apud Barros)**

### 2.2.2. Modelos de Desenvolvimento de Processo (ciclo de vida)

Atualmente existem diversos modelos para desenvolvimento de processos nas organizações, esses modelos, ao serem comparados aos modelos da engenharia de software, podem ser utilizados como ciclo de vida para gestão de melhoria de processos organizacionais.

Um bom projeto de implementação de processos precisa ter um planejamento bem estruturado com fases distintas, recursos e artefatos projetados. Entender que atividades fazem parte do processo e como cada atividade se relaciona umas com as outras, suas entregas e seus artefatos, é muito importante para assegurar qualidade na entrega final do projeto. Alguns modelos para projetos de processos são utilizados pelos gestores para obterem uma visão geral do projeto serão detalhados nessa seção.

### 2.2.2.1. PDCA

O ciclo PDCA (plan–do–check–act) é uma ferramenta utilizada para análise e melhoria de processos organizacionais e através de planos de ação, por ela utilizados, proporcionar melhoria na eficiência das equipes, bem como garantia em seus resultados. Foi idealizado na década de 20 por Walter A. Shewarth, e popularizado ao redor do mundo por William Edwards Deming que publicou e aplicou o método. Essa ferramenta é utilizada tanto para melhoria de processos quanto como um ciclo de vida de projetos de melhoria de processos. Rodrigues (2006, p.18) apresenta PDCA como:

> A melhoria contínua tem como suporte o controle e a otimização dos processos e foi a base para a Metodologia da Gestão da Qualidade Total. Através do Ciclo PDCA busca-se a monitoração dos processos produtivos para a melhoria contínua gradual (Kaizen), através da identificação e análise de resultados indesejáveis e da consequente busca de novos conhecimentos para auxiliar nas soluções

De acordo com Campos (1994), o PDCA é um ciclo proposto para propiciar a melhoria contínua e passa por 8 etapas conforme quadro abaixo:

**Tabela 1 - Etapas do PDCA proposto por Campos (1994) (fonte: a autora)**

| | | |
|---|---|---|
| **P – Plan – planejar** | 1 - Identificação do problema | A- Escolha do problema;<br>B- Histórico do problema;<br>C- Mostrar perdas atuais e ganhos viáveis;<br>D- Fazer análises;<br>E- Nomear responsáveis. |
| | 2 – Análise de Fenômeno | A- Descoberta das características do problema através de coleta de dados;<br>B- Descoberta das características do problema através de observação no local;<br>C- Cronograma, orçamento e meta. |
| | 3 - Análise do processo | A- Definição das causas influentes;<br>B- Escolha das causas mais prováveis (hipóteses);<br>C- Análise das causas mais prováveis (verificação de hipóteses). |
| | 4 - Plano de ação | A- Elaboração da estratégia de ação;<br>B- Elaboração do plano de ação para o bloqueio e revisão do cronograma e orçamento final. |

| | | |
|---|---|---|
| **D – Do – fazer** | 5 - Execução | A- Treinamento;<br>B- Execução da ação. |
| **C – Check – verificar** | 6 - Verificação | A- Comparação de resultados;<br>B- Listagem dos efeitos secundários;<br>C- Verificação da continuidade ou não do problema. |
| **A – Action – agir corretamente** | 7 - Padronização | A- Elaboração ou alteração do padrão;<br>B- Comunicação;<br>C- Educação e treinamento;<br>D- Acompanhamento da utilização do padrão. |
| | 8 - Conclusão | A- Relação dos problemas remanescentes;<br>B- Planejamento do ataque aos problemas remanescentes;<br>C- Reflexão. |

A Figura 11 apresenta graficamente o ciclo PDCA e suas etapas propostas por Campos (1994):


**Figura 11 - Ciclo PDCA – (fonte: http://www.indg.com.br/sobreoindg/metodopdca.asp, acesso em 06.out.2012)**

Marshall Junior et al (2006), apresenta fases do ciclo PDCA,da seguinte forma:

- 1ª Fase – Plan (Planejamento). Na fase de planejamento é fundamental que as metas do planejamento estratégico sejam determinadas, de maneira que simulem as condições do cliente e padrão de produtos, serviços ou processos. Dessa forma, as metas somente serão alcançadas, por meio das metodologias que contemplam as práticas e os processos.

- 2ª Fase – Do (Execução). Esta fase tem por objetivo a prática, por esta razão, é imprescindível oferecer treinamentos na perspectiva de viabilizar o cumprimento dos procedimentos aplicados na fase anterior.

- 3ª Fase – Check (Verificação). Nessa fase o que foi planejado é verificado mediante as metas estabelecidas e resultados alcançados.

- 4ª Fase – Act (Ação). A última etapa proporciona duas opções a ser seguida, a primeira baseia-se em diagnosticar qual é a causa raiz do problema bem como a finalidade de prevenir à reprodução dos resultados não esperados, caso, as metas planejadas anteriormente não forem atingidas. Já a segunda opção segue como modelo o esboço da primeira, mas com um diferencial se as metas estabelecidas foram alcançadas.

#### 2.2.2.2. DMAIC

A metodologia DMAIC (*Define, Measure, Analyse, Improve, Control*) é um ciclo de desenvolvimento de projetos de melhoria originalmente utilizado na estratégia Seis Sigma. Inicialmente, o DMAIC foi concebido para projetos relacionados à qualidade, porém como representa um ciclo organizado e ordenado de trabalho, ele também abrange projetos de aumento de produtividade, redução de custo. Esta metodologia é conhecida como uma das principais ferramentas para aprimoramento de processos, por isso é muito utilizada como guia em projetos de melhoria. Carvalho et al (2005), enfatiza que a metodologia DMAIC (definir, medir, analisar, melhorar e controlar), propor-se ao aprimoramento dos processos por meio da escolha destes e do melhoramento das pessoas a serem orientadas para alcançar os resultados tracejados.Para Aguiar (2006) , o DMAIC é uma metodologia de solução de problemas que tem como objetivo realizar melhorias em produtos, serviços e processos. Este autor define as fases do DMAIC da seguinte maneira:

- "D" Definir: definição do projeto, do assunto ao qual se deseja tratar. Nesta fase a equipe de trabalho é definida e são realizadas as primeiras reuniões para formalizar o projeto;

- "M" Medir: Nessa fase as métricas que estão ligadas ao projeto ou problema a ser estudado, são definidas e a coleta de dados é realizada;
- "A" Analisar: Nessa fase os dados coletados são analisados,;
- "I" Implementar: Nessa fase melhorias são propostas baseadas nos dados e análises realizadas nas fases anteriores;
- "C" Controlar: Nessa fase, deve ser verificado e certificado que as melhorias implementadas estejam sendo seguidas e efetivamente fazem parte do novo processo.

A Figura 12 ilustra o ciclo DMAIC e suas principais atividades de acordo com o *KAIZEN Institute*.

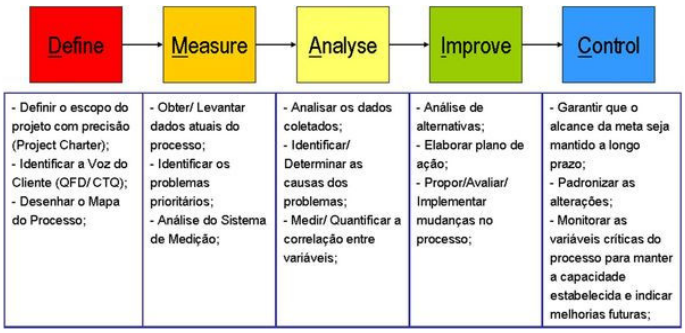



**Figura 12 – DMAIC – (fonte: http://br.kaizen.com/artigos-e-livros/artigos/dmaic.html, acesso em 06.out.2012)**

Para Aguiar (2006), o método DMAIC foi desenvolvido com apoio do PDCA e adota dimensão distinta, dependendo do seu uso, conforme ilustra a Figura 13.

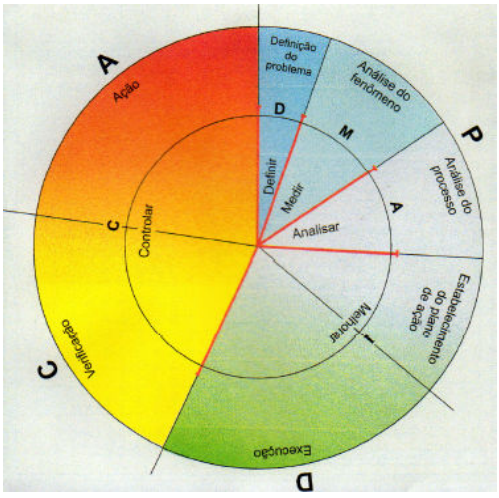



**Figura 13 - Comparação de do DMAIC de Melhorias com o PDCA de Melhorias. (fonte: Aguiar, 2006)**

#### 2.2.2.3. IDEF

A IDEF (*Integrated Computer Aided Manufacturing Definition*) é uma ferramenta de modelagem de processos que foi desenvolvida na década de 70 pela força aérea americana dentro do projeto ICAM (*Integrated Computer Aided Manufaturing*). A ferramenta apresenta o fluxo de informações dentro dos processos. Segundo Aguilar-Savén (2004), a família IDEF é utilizada de acordo com diferentes padrões e os mais conhecidos e usados para diferentes aplicações são sete conforme a seguir:

- IDEF0: usado para desenvolvimento estrutural e representação gráfica de processos ou sistemas empresariais complexos.
- IDEF1: usado para modelagem de informações, e captura de vistas conceituais das informações da empresa;
- IDEF1X: usado para modelagem de dados, e captura lógica das informações da empresa, sendo baseado em um modelo de relacionamento entre entidades;
- IDEF2: padrão para projeto de modelo de simulação usado para representar o comportamento da variação do tempo, em função dos sistemas de recursos de manufatura;
- IDEF3: usado para descrever e capturar aspectos comportamentais de um processo em diferentes pontos de vista do funcionamento da organização.
- IDEF4: padrão de projeto orientado a objetos que foi desenvolvido para apoiar projeto de implementação e aplicações em linguagem C;
- IDEF5: fornece uma teoria empírica fundamentada, que ajuda na criação, modificação e manutenção da ontologia;

Para a modelagem de processos de negócios, as versões mais utilizadas são o IDEF0 e o IDEF3. No IDEF0 o fluxo de informações existentes entre funções é mapeado, possibilitando uma visão gradativamente detalhada do processo com uma linguagem e semântica simples, permitindo representar todos os recursos existentes nos processos de negócio. Porém, Aguilar-Savén (2004) destaca no IDEF0 o fato das regras serem mais rígidas, o que facilita a modelagem e sua interpretação e de acordo com este autor, esta característica já não é encontrada em Fluxogramas, por exemplo. Já o IDEF3, ao contrário do IDEF0, caracteriza-se pelo fato dos eventos ou atividades serem descritos na

verdadeira ordem na qual estes ocorrem, levando em consideração as precedências temporais.

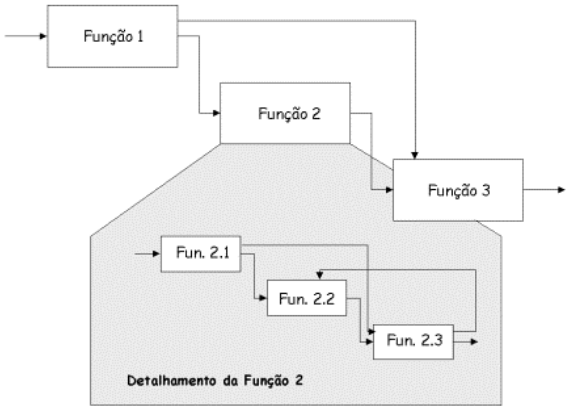



**Figura 14 – Esquema de detalhamento de funções - (fonte: http://www.numa.org.br/transmeth/ferramentas/ffmapeam.htm, acesso em 06.out.2012)**

#### 2.2.2.4. BPM

De acordo com o BPM CBOK (2010), a prática gerencial de BPM pode ser caracterizada como um ciclo de vida contínuo (processo) de atividades integradas de BPM. Projetos de processos de negócio são projetos que gerenciam desde o planejamento até a implantação e monitoramento dos processos de uma organização. O BPM CBOK (2010) sumariza o ciclo de vida de BPM, conforme ilustra Figura 15, como um conjunto gradual e interativo de atividades que incluem:

- Planejamento: Nessa fase são estabelecidas as estratégias e o direcionamento para o projeto de redesenho de processos. O ciclo de vida BPM começa com o desenvolvimento de um plano e uma estratégia dirigida a processos para a organização.

- Análise: Nessa fase várias metodologias podem ser incorporadas com a finalidade de entender os atuais processos organizacionais AS-IS no contexto das metas e objetivos desejados. A análise assimila informações oriundas de planos estratégicos, modelos de processo, medições de desempenho, mudanças

no ambiente externo e outros fatores, a fim de entender completamente os processos de negócio no escopo da organização como um todo.

- Desenho: Nessa fase é documentada a seqüência de atividades incluindo o desenho do trabalho realizado, em que tempo, em qual local, por quais atores de processo e utilizando qual metodologia. O desenho define o que a organização quer que o processo seja, e responde questões como: o quê, quando, onde, quem e como o trabalho ponta-a-ponta é realizado. Nessa fase outro ponto importante é assegurar que a definição de métricas e controles gerenciais apropriados estejam implementados para medição de desempenho e conformidade.

- Modelagem: Nessa fase o processo de negócio, bem como, os fatores ambientais que o habilitam ou restringem devem ser entendido. Para organizações que estão menos maduras na prática BPM, pode ser a primeira vez que o processo de negócio ponta-aponta tenha sido documentado.

- Monitoramento e Controle: A contínua medição e monitoramento de processos de negócio fornecem a informação necessária para que gestores de processo ajustem recursos, a fim de, atingir objetivos dos processos. No contexto do ciclo BPM, medição e monitoramento também provêem informações-chave de desempenho de processo através de métricas relacionadas às metas e ao valor para a organização.

- Transformação: Transformação de processo implementa o resultado da análise iterativa e o ciclo de desenho. Trata desafios de gerenciamento de mudança organizacional e está orientado à melhoria contínua e otimização de processo.

O ciclo de vida de BPM é aplicável tanto para projetos grandes (transformação de processos) quanto pequenos projetos (mudanças em rotinas ou atividades). Sendo que, para cada tipo de projeto as fases do ciclo de vida, de acordo com a complexidade e escala do projeto, poderão ter maiores detalhes, outros menos detalhes. Outro ponto importante a

observar é que existem fatores que podem impactar diretamente o ciclo de vida de BPM tais como organização, definição de processo, responsabilidade, patrocínio, medição, consciência, alinhamento, tecnologia da informação e metodologia BPM.

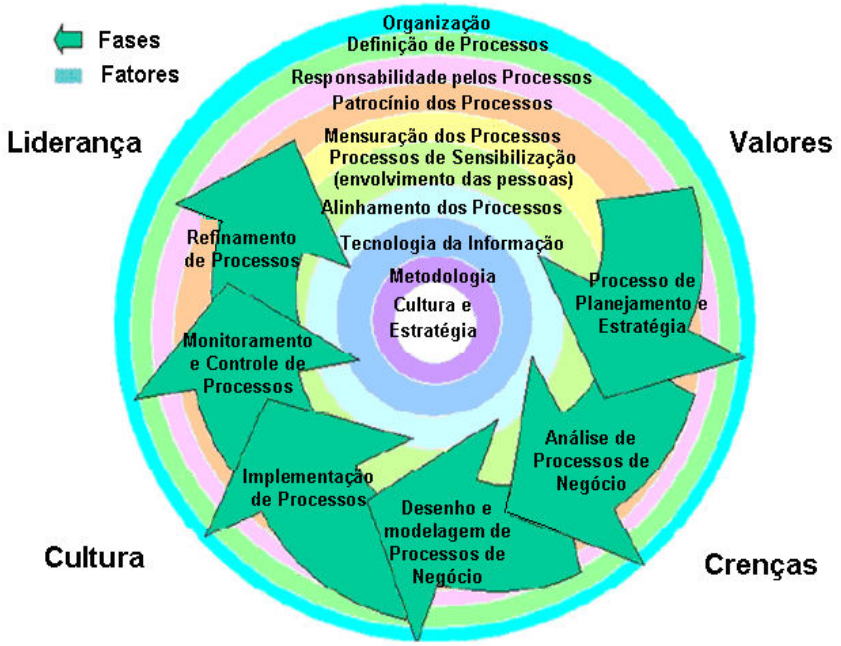



**Figura 15 - Ciclo de vida de BPM (Fonte: CBOK, 2010)**

## 2.3. Comparativo Engenharia de Software e Engenharia de Processos

Nos limites do estudo desse projeto, engenharia de software e engenharia de processos em muito se assemelham, e até mesmo se completam, tanto quando tratado projeto, quanto ciclo de vida, maturidade, gestão, recursos, custo entre outros. Diante desse cenário e com o intuito de aprofundar nas diversas técnicas de estimar projetos, buscando àquelas que podem contribuir para projetos de processos, nessa seção, serão tratados alguns pontos em comum entre essas duas áreas. Logo, primeiramente será necessário compreender melhor sobre projetos de processos, o que são projetos, o que são processos e o que é gerenciamento de processos de negócio (BPM).

### 2.3.1. Projetos, Processos e Gerenciamento de Processos de Negócio (BPM)

De acordo com o PMBOK (2008), um projeto pode ser definido, em termos de suas características distintivas, como sendo "um empreendimento temporário feito para criar um produto ou serviço único". Tuman (1983) em uma definição mais completa descreve projeto como "uma organização de pessoas dedicadas visando atingir um propósito e objetivo específico". O autor ainda afirma que "projetos geralmente envolvem gastos, ações únicas ou empreendimentos de altos riscos no qual tem que ser completado numa certa data por um montante de dinheiro, dentro de alguma expectativa de desempenho. No mínimo todos os projetos necessitam de terem seus objetivos bem definidos e recursos suficientes para poderem desenvolver as tarefas requeridas".

A definição de processos de acordo com o Guia para gerenciamento de Processos de Negócio – BPM CBOK (2010, p.23) é:

> .. um conjunto definido de atividades ou comportamentos executados por humanos ou máquinas para alcançar uma ou mais metas. Os processos são disparados por eventos específicos e apresentam um ou mais resultados que podem conduzir ao término do processo ou a transferência de controle para outro processo. Processos são compostos por várias tarefas ou atividades inter-relacionadas que solucionam uma questão específica. No contexto do gerenciamento de processos de negócio, um "processo de negócio" é definido como um trabalho ponta-a-ponta que entrega valor aos clientes. A noção de trabalho ponta-a-ponta é chave, pois envolve todo o trabalho cruzando limites funcionais necessários para entregar valor aos clientes.

Logo, projetos de processos de negócio são projetos que gerenciam o ciclo de vida desde a o planejamento até a implantação e monitoramento dos processos de uma organização. Sendo assim, os projetos de processos (Engenharia de Processos) muito se assemelham aos projetos de software (Engenharia de Software), pois ambos possuem tem um ciclo de vida específico, um objetivo, uma equipe definida, um planejamento de custo, planejamento de mitigação para riscos e todos visam implantação com qualidade e no tempo programado. Em ambos os tipos de projetos, a fase de planejamento é muito importante e envolve compreender a forma de metrificar e estimar os prazos, o custo e o escopo dos projetos. Dessa forma, o estabelecimento de métricas e estimativas, constitui uma das principais atividades do planejamento de ambos os tipos de projeto e em ambas as áreas – Engenharia de Software e Processos. Para melhor compreensão, na Tabela 2 abaixo, são

apresentadas algumas características comuns entre as áreas da Engenharia de Software e Engenharia de Processos, principalmente entre os projetos e processos a elas referenciados.

**Tabela 2 - Comparação entre Engenharia de Software e Engenharia de Processos**

| Itens Comparativos | Engenharia de Software | Engenharia de Processos |
|---|---|---|
| | **Projetos** | **Processos** |
| Objetivo | .. um empreendimento temporário feito para criar um produto ou serviço único (PMBOK, 2008) | .. um conjunto definido de atividades ou comportamentos executados por humanos ou máquinas para alcançar uma ou mais metas ( BPM CBOK, 2010). |
| Gerenciamento | .. aplicação de conhecimento, habilidades, ferramentas e técnicas às atividades do projeto a fim de atender aos seus requisitos. (PMBOK, 2008) | ... uma abordagem disciplinada para identificar, desenhar, executar, documentar, medir, monitorar, controlar e melhorar processos de negócio de forma a alcançar resultados consistentes e alinhados com as metas estratégicas de uma organização (BPM CBOK, 2010). |
| Ciclo de Vida | .. é uma representação abstrata de um processo para explicar as diferentes abordagens de desenvolvimento (Sommerville, 2003).<br>(1) Iniciação;<br>(2) Planejamento;<br>(3) Execução;<br>(4) Controle;<br>(5) Encerramento. | um conjunto gradual e interativo de atividades que incluem:<br>(1) Planejamento;<br>(2) Análise;<br>(3) Desenho e Modelagem;<br>(4) Implantação;<br>(5) Monitoramento e Controle; e<br>(6) Refinamento. |
| Modelos de Maturidade | ... grau em que uma organização aplica as boas práticas de gerenciamento de projetos em cada um dos domínios estabelecidos: projetos, programas e portfólio (Kerzner, 2003).<br>Exemplo: Capability Maturity Model Integration (CMMI) que organiza os níveis de maturidade | Estrutura de maturidade em níveis para formar alicerces sucessivos à melhoria<br>contínua de processos. Compreende um conjunto de objetivos de processo que, quando satisfeito, estabilizam um componente importante de processo. Os níveis de maturidade são: |

| | | |
|---|---|---|
| | da seguinte forma:<br>(1) Inicial;<br>(2) Gerenciado;<br>(3) Definido;<br>(4) Quantitativamente gerenciado;<br>(5) Otimizado. | (1) Inicial<br>(2) Gerenciado<br>(3) Padronizado<br>(4) Previsível<br>(5) Inovando |
| Método de Estimativa | Existência de várias técnicas para estimar projetos de software. | Durante a escrita desse projeto, ainda não foi identificada nenhuma publicação de estudos relacionados a técnicas específicas para projetos de processos. |

### 2.3.2. Técnicas para estimar projetos

Ao longo dos anos algumas técnicas para estimar projetos foram implementadas e várias são utilizadas até hoje para desenvolvimento de software, tais como:

#### 2.3.2.1. Estimativa de Putnam:

O modelo de Putnam (1978) relaciona o número de linhas de código ao tempo e esforço de desenvolvimento. É um modelo dinâmico de múltiplas variáveis que pressupõe uma distribuição do esforço específico ao longo da existência de um projeto de desenvolvimento de software. De acordo com Pillai et al (1997), o modelo pode ser utilizado na análise de processos para avaliar o impacto de um cronograma apertado e prever custos a partir da evolução do projeto. Eles ainda afirmam que o modelo prevê, de forma simples, uma estimativa de custos para o desenvolvimento de um software, porém sua capacidade em prever tempo de desenvolvimento e requisitos para quantidade de mão-de-obra em um estado inicial do desenvolvimento não são satisfatórios e isso se deve ao fato que uma previsão inicial (early prediction) falha, pois modelos lineares simplificados falham em capturar a variação inicial das observações e segundo eles, existem duas formas de tentar resolver essa questão: ajustar a curva com um grau mais elevado para o conjunto de dados e/ou encontrar uma alternativa de transformação para os eixos. Essas

opções, no entanto, se aplicadas ao modelo de Putnam, criariam uma complexidade enorme ao modelo e o tornariam muito mais dependente dos dados e não genérico. Abaixo segue a equação que o modelo utiliza para o desenvolvimento de softwares:

**L = Ck.K1/3.td4/3, onde**

**L** = linha de código
**Ck** = constante de estado de tecnologia

- Ck = 2.000 –ambiente de desenvolvimento pobre
- Ck = 8.000 – métodos em prática, documentação e revisões adequada
- Ck = 11.000 – ambiente ótimo (ferramentas e técnicas automatizadas)

**K** = esforço (pessoa – ano)
**Td** = tempo de desenvolvimento (anos)

Novo arranjo: Esforço – K = L3/(ck3.td4)

Para aplicação desta técnica, para projetos de processos, deve-se levar em consideração que o modelo verifica os desvios que podem acontecer ao longo dos projetos e taxa de erro nas estimativas e isso pode garantir um tratamento para uma margem de erro controlada nos projetos. Porém a equação proposta por Putnam (1978) leva em consideração linhas de código, constante de estado de tecnologia e ferramentas automatizadas. Sabe-se que projetos de processos nem sempre são automatizados, logo podemos chegar à conclusão que a equação, a princípio, não atenderia em sua totalidade a um projeto de processos.

#### 2.3.2.2. Cost Constructive Model (COCOMO)

O COCOMO foi apresentado por Boehm (1981), tendo sido construído e calibrado inicialmente a partir de informação de um número considerável de projetos concluídos, em torno de 63 projetos, em sua maior parte da empresa TRW Systems, Inc. Pelo fato deste método ter evoluído, sua primeira versão passou a ser chamada de COCOMO 81.

O COMOMO 81 de acordo com Boehm (1981) é um método que tem como premissa básica o tamanho pré-definido do projeto, ou seja, uma dimensão do projeto. O método busca estimar esforço, prazo, custo e tamanho de equipe, necessários ao desenvolvimento do software. Ele considera três modos de desenvolvimento e é

apresentado na forma de um conjunto de modelos divididos hierarquicamente em três estágios distintos conforme a seguir:

- Modelo Básico => modelo estático de valor simples que computa o esforço do desenvolvimento de software e o custo em função do tamanho de programa expresso em linhas de código estimadas, de acordo com Abreu (1998). Esta versão é aplicável à grande maioria dos projetos de software, de pequeno ou médio porte, porém é considerada limitada por não considerar fatores que interferem no desenvolvimento do projeto, como restrições de hardware, qualificação e experiência do pessoal de desenvolvimento e uso de ferramentas técnicas modernas, entre outros.

- Modelo Intermediário => calcula o esforço de desenvolvimento de software em função do tamanho do programa e de um conjunto de direcionadores de custo, alternativamente chamados atributos ou fatores de software, que incluem avaliações subjetivas do produto, do hardware, do pessoal e dos atributos do projeto. De acordo com Abreu (1998), corresponde a considerar a influência de um conjunto de vários fatores, relativos quer ao sistema a produzir (produto) propriamente dito, quer ao suporte computacional (tecnologia utilizada), fator humano e organização do processo de desenvolvimento de software. A influência destes fatores, em número de 15 no modelo originalmente proposto, deve ser avaliada numa escala discreta e ponderada.

- Modelo Completo => incorpora todas as características da versão intermediária, porém em cada passo do processo de engenharia de software, além de adicionar aspectos como a decomposição de um sistema de grande dimensão em subsistemas. Outros aspectos correspondem à distribuição das estimativas de 33 esforço e de prazo por fase e por atividade e à influência diferenciada de cada fator influenciador do custo por fase. De acordo com Aguiar (2004) O COCOMO II foi e continua sendo desenvolvido em uma universidade, a University of Southern California (USC) devido a isso é suportado por uma ferramenta de software gratuita, disponibilizada pela própria USC. O autor também enfatiza que devido ao projeto possuir parceria

com a Rational o modelo é totalmente compativel com o Rational Unified Process (RUP).

**Classificação do produto**

A aplicação do COCOMO 81 começa pela classificação do produto a ser mensurado, categorizando o software em um de três tipos fundamentais de desenvolvimento identificados por Boehm (1981):

- Modo orgânico - aplicável a ambientes de desenvolvimento estáveis, com pouca inovação e a projetos com equipes de dimensão relativamente pequena.

- Modo semidestacado – também chamado de modo difuso, representa o modo intermediário entre os modos orgânicos e embutido. Aplicável a projetos com características tanto do modo orgânico quanto do modo embutido

- Modo embutido – também conhecido como modelo restrito, aplicável no desenvolvimento de sistemas complexos embutidos em hardware, com muita inovação, com restrições severas e/ou com requisitos muito voláteis.

### 2.3.2.3. COCOMO II

De acordo com Aguiar (2004), devido à idade dos projetos que embasaram o modelo COCOMO 81, assim como sua incapacidade de lidar com ciclos de vida iterativos e com a utilização de componentes *Commercial-Off-The-Shelf* (COTS), o COCOMO 81 é atualmente considerado obsoleto, tendo sido substituído por sua versão II, publicada em 2000.

Em um estudo realizado por Trindade, Pessoa e Spinola (1999) são apresentadas as diferenças entre o COCOMO 81 e o COMOMO II comparando suas características básicas.

| | COCOMO 81 | COCOMO II |
|---|---|---|
| Estrutura do Modelo | Modelo tríplice, categorizador exclusivo, onde o usuário qualifica seu projeto em uma e apenas uma das opções:<br>• Básico;<br>• Intermediário;<br>• Avançado. | Modelo tríplice, onde o usuário progride, por exigências de arquitetura e redução de riscos, em um desenvolvimento do tipo espiral, por:<br>• Composição da aplicação; OU<br>• Pré-Projeto;<br>• Pós-Arquitetura. |
| Fórmulas matemáticas para as equações | No cálculo do esforço:<br>$E_{(H/M)} = A \times Tamanho^{B} \times Ci$<br>No cálculo do prazo:<br>$P_{(M)} = J \times E^{K}$ | No cálculo do esforço:<br>$E_{(H/M)} = A \times Tamanho^{B} \times Ci + F_{re\text{-}eng}$<br>No cálculo do prazo:<br>$P_{(M)} = J \times E^{K}$ |
| Parâmetros | O parâmetro **A**:<br>• Orgânico **2,4** (bás) ou **3,2**<br>• Restrito (int/Avanç)<br>• Difuso **3,0** (sempre)<br>**3,6** ou **2,8**<br>O parâmetro **J = 2,5** | Os parâmetros A e J são únicos, para todos os submodelos, mas não são estáticos, ou seja, podem mudar quando uma nova calibração foi executada.<br>Na calibração da versão 1999.0:<br>**A = 2,94** **J = 3,67** |
| Tamanho do software | Instruções-fonte entregues (DSI) | Pontos de objetos, Pontos de função ou Linhas de código-fonte (SLOC) |
| Expoentes | O COCOMO 81 trabalha com constantes fixas selecionadas pela categoria do software, segundo sua dimensão e o tipo de aplicação que se possa fazer dele:<br>• Orgânico **B=1,05** **K=0,38**<br>• Restrito **B=1,12** **K=0,35**<br>• Difuso **B=1,20** **K=0,32** | O COCOMO II, no cálculo do esforço, estabelece o expoente B baseado na análise dos valores possíveis em cinco fatores de equilíbrio (Fe):<br>• **PREC**, Precedência;<br>• **FLEX**, Flex. De Desenvolvimento;<br>• **RESL**, Res. Arquitetura ou Risco;<br>• **TEAM**, Coesão Da Equipe;<br>• **PMAT**, Maturidade do Processo,<br>através da fórmula: $B = \alpha + 0,01 \times \sum Fe$.<br>O parâmetro α é alterado por calibrações.<br>Na versão 1999.0, $\alpha = 0,91$<br>No submodelo Composição da aplicação, **B = 1** (uma constante)<br>No cálculo do prazo, o expoente K é calculado pela fórmula: $K = \beta + 0,2 \times (B - 1,01)$.<br>O parâmetro β é alterado por calibrações.<br>Na versão 1999.0, $\beta = 0,28$ |
| Direcionadores de custo ($C_i$) | 15 direcionadores são utilizados nos submodelos difuso e restrito (o submodelo orgânico não prevê ajustes):<br>• **RELY** Confiabilidade<br>• **DATA** Tamanho da B.D.<br>• **CPLX** Complexidade do software<br>• **TIME** Restrições do tempo<br>• **STOR** Restr. ao uso de memória<br>• **VIRT** Mudanças do ambiente<br>• **TURN** Tempo de resposta | Diversos direcionadores são utilizados com conjugações diferentes em submodelos diferentes:<br>submodelo Pré-Projeto:<br>• **RCPX** Conf.+ complex. do softw.<br>• **RUSE** Reusabilidade requerida<br>• **PDIF** Dificuldades c/ plataforma<br>• **PERS** Capacidade do pessoal<br>• **PREX** Experiência profissional<br>• **FCIL** Instalações |

| | | |
|---|---|---|
| | • **ACAP** Capacidade dos analistas<br>• **AEXP** Experiência na aplicação<br><br>• **PCAP** Cap. Dos programadores<br>• **VEXP** Experiência com hardware<br>• **LEXP** Experiência c/ linguagem<br>• **MODP** Técn. modernas de progr.<br>• **TOOL** Uso de ferramentas de soft.<br>• **SCED** Prazo requerido | • **SCED**<br>submodelo Pós-Arquitetura:<br>• **RELY**<br>• **DATA**<br>• **CPLX**<br>• **RUSE** Reusabilidade requerida<br>• **DOCU** Documentação<br>• **TIME**<br>• **STOR**<br>• **PVOL** Mudanças de plataforma<br>• **ACAP**<br>• **AEXP**<br>• **PCAP**<br>• **PEXP** Experiência com plataforma<br>• **LTEX** Exp. c/ lingu. e ferramentas<br>• **PCON** Continuidade de pessoal<br>• **TOOL**<br>• **SITE** Desenvolvimento multi-local<br>• **SCED** |
| Fator de re-engenharia de software ($F_{re-eng}$) | No COCOMO 81, nada consta | No COCOMO II, o cálculo do esforço extra necessário, quando um processo de reforma de sistema está em andamento, alterando algoritmos, com ou sem trocas de linguagens, é dado pela fórmula:<br>$F_{re-eng} = \left[ \frac{ASLOC.\left(\frac{AT}{100}\right)}{ATPROD} \right]$<br>Em processos de construção de um produto software novo, $F_{re-eng} = 0$ |
| Outras diferenças | Modelo fundamentado sobre:<br>• Fórmula de regressão, linear;<br>• Suposição de requerimentos razoavelmente estáveis. | Possui muitos melhoramentos, incluindo:<br>• Fórmula de regressão, não linear;<br>• Modelo de reuso que visa esforços necessários ao entendimento e à assimilação;<br>• Avaliação das interrupções do processo, as quais são utilizadas para endereçar volatilidades de requerimentos;<br>• Características de autocalibração. |

tabela 1 - Quadro comparativo entre características básicas dos dois modelos COCOMO. Adaptada de Clark (1997) e Reifer & Outros (1998).

**Figura 16 - Comparação entre características básicas dos dois modelos COCOMO - extraído de Trindade, Pessoa e Spinola (1999)**

O COCOMO considera três modos de desenvolvimento e é apresentado na forma de um conjunto de modelos divididos hierarquicamente em três estágios distintos. É possível que estes modos possam ser reaproveitados e adaptados para projetos de processos. Porém tanto o COMOMO 81, quanto o COCOMO II buscam medir esforço, prazo, tamanho e custo, necessário para o desenvolvimento de software tendo como premissa básica a dimensão do projeto. Logo, para a

utilização em projetos de processos será necessário primeiramente adequar uma técnica capaz de mensurar a dimensão do projeto.

#### 2.3.2.4. LOC (linha de código)

De acordo com Cappelli et al (2009), LOC (Lines of Code) é uma métrica usada para medir o tamanho de um programa ou sistema usando o número de linhas de código que refe-se ao número de expressões executáveis desconsiderando-se comentários, quebras de linhas, etc. De acordo com Thammarak (2010), LOC (Lines of Code) é uma das formas mais simples de medir e averiguar a complexidade ou tamanho de um programa. De acordo com esse autor, é possível utilizar essa métrica para projeto de processos de negócio, de maneira que uma atividade de processos pode ser vista como equivalente a uma linha de código. Assim, a aplicação de LOC para BPM pode ser considerada a forma mais simples e fácil de medir o tamanho de um modelo de processo de negócio. O autor ainda cita como exemplo, três formas de subdividir a forma de contar: número de atividades de um processo (NOA), número de atividades e controle de fluxo de componentes em um processo (NOAC), e número de atividades, associações e divisões de um processo (NOAJS). Além disso, pode-se ainda utilizar o número de chamadas recebidas em um processo, ou seja, contar o número de todas as entradas em um determinado processo e o número de saída de um determinado processo. Porém os autores Cappelli et al (2009), citam que LOC não leva em consideração a estrutura do modelo, pois um modelo pode ser desenhado de forma estruturada e de fácil entendimento ou, de forma oposta, ser desestruturado e de difícil entendimento.

#### 2.3.2.5. MCCABE-METRIC

De acordo com Cappelli et al (2009), o número ciclomático, introduzido por McCabe, é uma das métricas de software mais difundidas e usadas. Corresponde ao numero de decisões binárias mais 1. Decisões não binárias com N resultados possíveis são

contadas como N-1 decisões binárias. De acordo com Pressman (2001) a métrica foi desenvolvida em 1976 e se baseia numa representação do fluxo de controle de um programa, na complexidade ciclomática de um gráfico de programa para um módulo. Ela mede o número de caminhos linearmente independentes num módulo de programa. Logo, número de regiões é computado contando-se todas as áreas delimitadas e a área não delimitada fora do gráfico. A Figura 3 tem cinco regiões (anotadas como R1 a R5) e, dessa forma, tem uma métrica de complexidade ciclomática igual a 5.

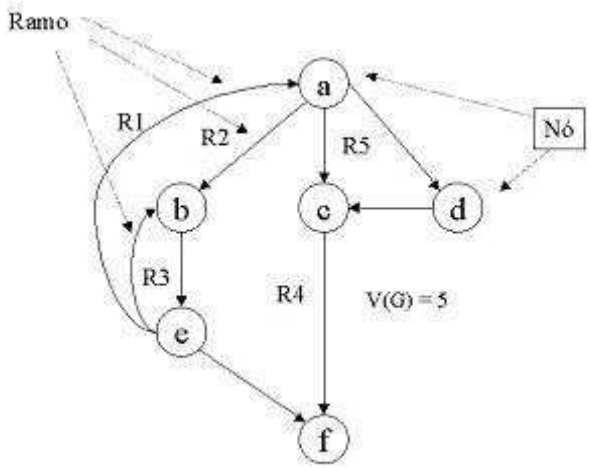



**Figura 17 - Complexidade Ciclomática – McCabe - extraído de Pressman (2001)**

De acordo com a Tabela 1 de Mccabe e Watson (1994), quando maior o valor da complexidade ciclomática, maior será a avaliação do risco

**Tabela 3- Avaliação da Complexidade Ciclomática de McCabe**

| Complexidade Ciclomática | Avaliação de Risco |
|---|---|
| 1 -10 | Um programa simples, sem muito risco |
| 11-20 | Mais complexo, risco moderado |
| 21-50 | Complexo, programa de alto risco |
| > 50 | Programa intestável (risco muito alto) |

De acordo com Thammarak (2010), McCabe calcula o número de caminhos linearmente independentes em um programa. Para cada módulo é calculado E- N + 2, onde E é o número de arestas que representam a transferência de comando entre nós

(caminhos tanto de entrada quanto de saída) e N é o número de nós. Por exemplo, se um programa tem 14 arestas e 11 nós, então este programa tem complexidade é 14-11 2 = 5. Este programa pode ser considerado de fácil compreensão e de fácil manutenção de acordo com a tabela 1. O autor descreve que essa métrica pode ser aplicada diretamente para medir projetos de processos de negócio para isso cada nó representaria cada atividade do processo e as arestas os fluxos de trabalho.

#### 2.3.2.6. USE CASE ESTIMATION

De acordo com o estudo de Cappelli et al (2009), a técnica de Pontos por Caso de Uso (UCP) se baseia na identificação e classificação de atores e casos de uso de um projeto de software para estimar o esforço necessário para o desenvolvimento deste projeto.

De acordo com Aguiar (2003), os pontos por caso de uso foram criados por Gustav Karner em 1993 como uma adaptação específica aos Pontos de Função. Apesar de pouco divulgado, os Pontos por Caso de Uso (UCP) tem sido estudados por vários pesquisadores no meio acadêmico e na indústria. De acordo com Aguiar (2003), um estudo de 2001, feito por Bente Anda, indicou que a variação nos estilos dos casos de uso pode impactar a quantidade de UCP obtida através do método, mesmo assim afirma que é possível utilizar os UCP na estimativa de esforço para alguns projetos. De acordo com Garcia (2000) os casos de uso descrevem um comportamento do sistema sob diversas condições e que produza um resultado observável que tem algum valor para um ator que interage com o sistema. Garcia (2000) apresenta de forma sucinta como a captura dos casos de uso são interligadas aos processos de negócio da organização. Logo, se os processos de negócio foram bem definidos, os casos de uso possuirão um refinamento mais apurado. Sendo assim, o autor propõe um novo modelo, onde os casos de uso são organizados em dois níveis, um nível onde cada processo de negócio é associado com um caso de uso e um segundo nível onde a partir dos casos de uso interligados aos processos de negócios, é realizado um novo agrupamento. A proposta possui cinco passos seqüenciais:

1. A identificação e delimitação dos processos de negócios dentro da organização.
2. O mapeamento das funções envolvidas nos processos de negócio, e seu desenho em um modelo que descreve as interações entre os agentes da organização durante a execução de um caso de uso.
3. A modelagem do fluxo de trabalho de cada processo de negócio e a descrição das regras de negócio que limitam os processos.
4. A extração dos casos de uso do sistema a partir das atividades que compõem os processos de negócio.
5. O estabelecimento do modelo conceitual de dados (objetos de informação) no diagramas de atividades.

Todos os elementos criados durante a modelagem são especificados em um glossário.

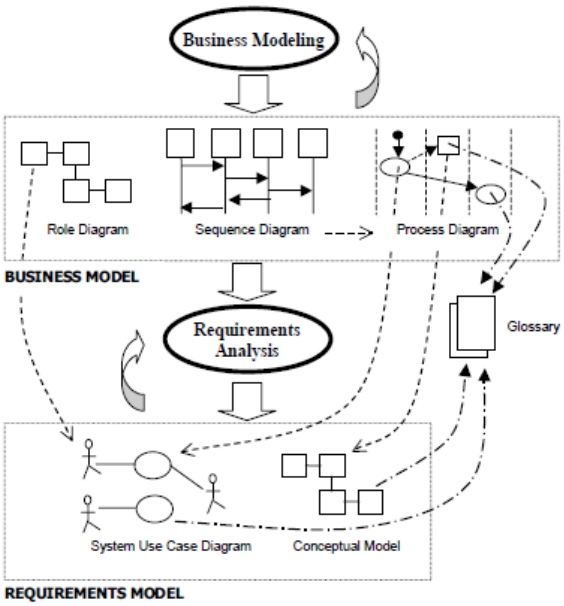



**Figura 18 – Modelo de Rastreabilidade entre os processos de negócio e os requisitos – extraído de Garcia (2000)**

De acordo com Thammarak (2000) os agentes dos casos de uso muito se assemelham aos agentes de processos, por exemplo, as atividades dos casos de uso podem representar as atividades de um processo de negócio. Os atores ou agentes do processo de negócio podem ser os mesmos agentes ou atores dos casos de uso. A descrição dos casos de uso pode ser um detalhamento de uma atividade dos processos de negócio da mesma

forma o detalhamento das regras de negocio e dos requisitos não funcionais. Thammarak (2000) detalha que não foi encontrada uma métrica direta para mensurar os processos, mesmos que estes estejam vinculados aos processos de negócio. No entanto, este autor orienta que se pode adotar as métricas existentes para medição por caso de uso para os processos de negócio.

# CAPITULO 3

# METODOLOGIA

## 3.1. Metodologia e Condução do Experimento

O tipo de pesquisa será descritiva (qualitativa) e os métodos e técnicas utilizados para pleno atendimento aos objetivos desse projeto serão:

**Tabela 4 - Alinhamento dos Objetivos com a Metodologia proposta**

| Objetivos do Projeto | Métodos e Técnicas |
|---|---|
| a) Analisar técnicas e estimativas de processo de software. | Pesquisa bibliográfica e análise documental, onde serão analisadas técnicas e estimativas de processo de software para entendimento de como estas podem agregar valor em medição de projetos de software. |
| b) Desenvolver um método para estimar esforço para projetos de redesenho de processos. | Realização de Experimento: Nessa etapa será desenvolvido um método para estimar esforço para projetos de processos. |
| a) Verificar e validar a aplicabilidade do modelo proposto. | A aplicabilidade do modelo proposto será analisada realizando testes e ajustes necessários no método através da utilização de variável controlada. |
| b) Analisar se os valores estimados para os projetos correspondem aos valores reais (esforço realizado). | Análise do resultado do experimento. Nessa etapa serão comparadas as estimativas de projetos finalizados passados com os valores resultantes do método proposto.<br><br>Variável independente = série histórica com os dados de projetos passados (previsto e realizado). |

O experimento será realizado em organizações que já identificaram a necessidade de desenhar ou redesenhar seus processos organizacionais por analistas que já trabalharam pelo menos com uma técnica ou método de mensuração de projetos. Entende-se que para melhor compreensão do estudo, este trabalho deve seguir uma metodologia fundamentada e utilizada por órgãos reconhecidos mundialmente. Logo este estudo seguirá o modelo da o *Practical Software Measurement* (PSM) realizando adaptações quando necessário.

De acordo com o *Practical Software Measurement* (PSM), o modelo estrutura a atividade de mensuração em um projeto de software, e é um processo para medir e melhorar processos, projetos e produtos de software. De forma prática o PSM procura resolver dois problemas:

- Como especificar de uma maneira formal as medidas que deverão ser utilizadas no projeto, ou seja, como especificar formalmente os indicadores a serem usados;
- Como deverá ser conduzido o processo de medição.

Para atingir tais objetivos, o PSM faz uso de dois modelos:

- Modelo de Informação: que fornecer um caminho para a seleção das medidas utilizadas;
- Modelo de Processo: que serve de guia a implementação do PSM.

Após a construção do método para estimar projetos de processos, baseados no referencial teórico objetivo deste estudo, este trabalho seguirá o modelo de Processo como guia para implementação de tal ferramenta e assim garantir qualidade em sua entrega. O Modelo de Processo do PSM orienta o praticante na condução das atividades de medição em um projeto de desenvolvimento de software. Neste trabalho, o modelo será adaptado para condução do projeto dessa dissertação, onde será criado um método para estimar (estimativa) projetos de processos, além da gestão da própria medição, ou seja, da aplicação de tal estimativa criada.

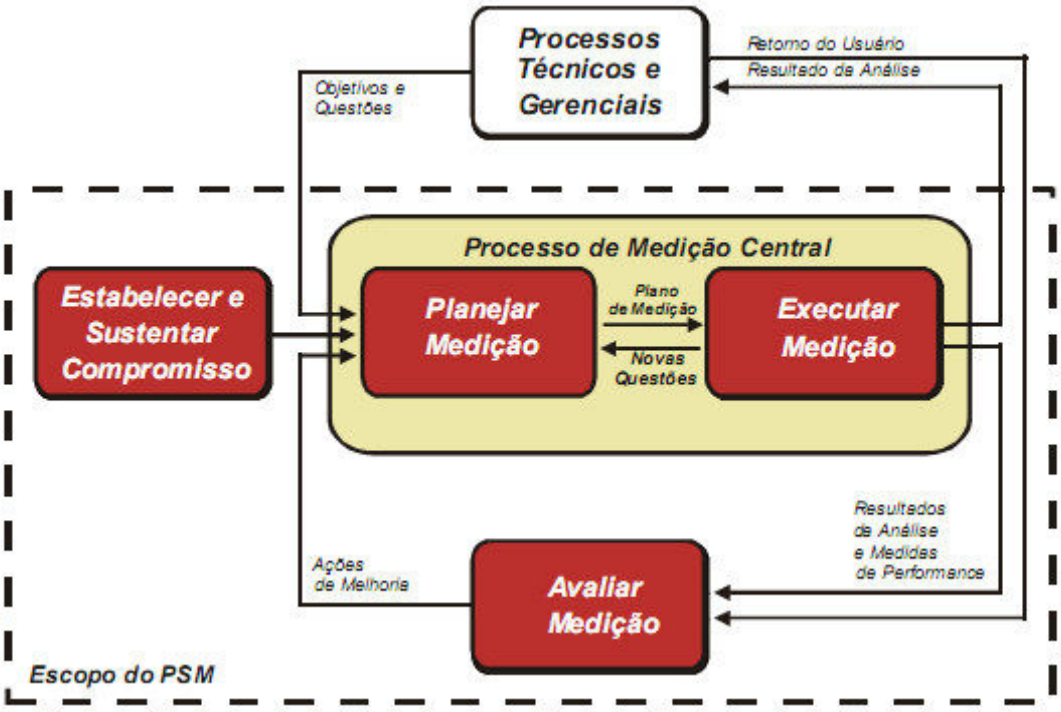



**Figura 19 - Modelo de processo de medição PSM - (Fonte: Modelo de Processo do PSM. Adaptado de Borges [Borges, 2003] e McGarry e outros [McGarry et al., 2002].**

O modelo, descreve quatro principais fases de acordo com McGarry et al., (2002):

1. Planejar Medição (*Taillor Measures):* envolve a identificação das necessidades de informação de um projeto e a seleção das medidas mais adequadas para atender a essas necessidades;
2. Executar Medição (*Apply Measures*): envolve a execução do modelo de informação para produzir dados necessários para suportar as etapas de decisão de uma organização;
3. Avaliar Medição (*Evaluate Measurement*): envolve a aplicação de técnicas de medição e análise para avaliar o próprio processo de mensuração;
4. Estabelecer e Sustentar Compromisso (*Implement Process*): assegura que a mensuração é apoiada tanto em nível operacional quanto em nível gerencial.
5. Adequando a este estudo temos:

**Tabela 5 - Fases do PSM e sua aplicação a este projeto (Fonte: a autora)**

| Fases | PSM | Adequação do PSM VISÃO MACRO |
|---|---|---|
| Planejar Medição (Taillor Measures): | Envolve a identificação das necessidades de informação de um projeto e a seleção das medidas | Identificar as técnicas e/ou métodos para estimar projeto s de software capazes de atender aos |

| | | |
|---|---|---|
| | mais adequadas para atender a essas necessidades; | projetos de processos para assim conduzir o experimento. |
| Executar Medição (Apply Measures): | Envolve a execução do modelo de informação para produzir dados necessários para suportar as etapas de decisão de uma organização; | Criação de um método para estimar projetos de processos baseado nos métodos e/ou estimativas estudadas. |
| Avaliar Medição (Evaluate Measurement): | Envolve a aplicação de técnicas de medição e análise para avaliar o próprio processo de mensuração; | Realização de testes com a estimativa proposta, em projetos de processos já finalizados cujas medições são reais. (Variável Controlada).<br><br>Comparar com as estimativas previstas e realizadas em projetos finalizados, criadas sem a utilização da nova estimativa (Variável Independente). |
| Estabelecer e Sustentar Compromisso (Implement Process): | Assegura que a mensuração é apoiada tanto em nível operacional quanto em nível gerencial. | Realizar melhorias sugeridas pela equipe operacional e gerencial criando pontos de calibração. |

De acordo com o guia PSM cada fase envolve atividades distintas que poderão auxiliar no experimento objetivo deste trabalho, sendo eles:

**Tabela 6 - Fase Planejar Medição - PSM e sua aplicação no projeto (Fonte: a autora)**

| Fase | Atividades | Adequação do PSM |
|---|---|---|
| Planejar Medição (Taillor Measures): | **Identificar e Priorizar Necessidades de Informação.**<br><br>Nessa atividade o PSM busca identificar as necessidades de informação dos projetos. As fontes para obtenção dessas informações são: avaliações de risco, suposições e restrições do projeto, utilização de tecnologias específicas, critérios de aceitação do produto, requisitos externos e experiências anteriores. | A motivação central deste trabalho é criar uma estimativa para projetos, capaz de atender as necessidades gerais dos processos de processos, também atendendo seus pontos específicos e características distintas. Nessa atividade serão levantados as características e os pontos específicos para atender a projetos de processos. |
| | **Selecionar e Especificar Medidas.**<br>Nessa atividade são selecionadas as medidas básicas, medidas derivadas e indicadores que virão a ser utilizados no atendimento às necessidades de informação anteriormente estabelecidas. | Nessa atividade serão avaliadas as estimativas e/ou métodos com pontos positivos para a condução do experimento.<br>Também nessa atividade, um protótipo do método, será criado. |
| | **Integrar Mensuração aos Processos do Projeto.** Basicamente nessa atividade será examinado como os dados serão coletados e analisados, a fim de satisfazer as necessidades de informação do projeto. | Nessa atividade será verificado juntamente aos gerentes de projetos em quais etapas de um projeto de processos o método para estimar deverá ser aplicado e a forma de coleta de informação. |

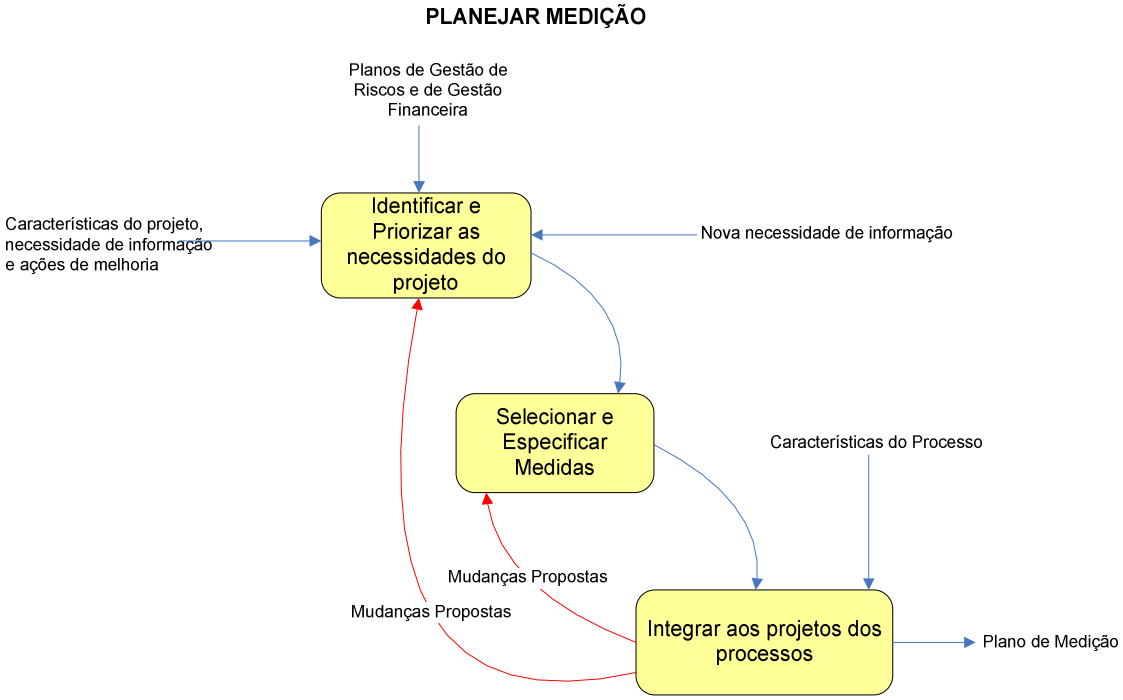



**Figura 20 - Fases da Macro-Atividade Planejar Medição - extraído de PSM Measurement Process (PSM GUIDE 4.0B.PART 1.1) e adaptado pela autora**

**Tabela 7- Fase Executar a Medição – PSM e sua aplicação no projeto (Fonte: a autora)**

| Fase | Atividades | Adequação do PSM |
|---|---|---|
| Executar Medição (Apply Measures): | **Coletar e Processar Dados** | Após a coleta e análise de todas as informações necessárias para a realização da estimativa, o método será criado e o experimento será aplicado nos projetos nas etapas definidas pelos gerentes de projeto. |
| | **Analisar Dados** | Os dados gerados pelo experimento serão comparados com as estimativas previstas e as realizadas, comparando o nível de assertividade com o real realizado, ou seja, gerando os indicadores de assertividade. |
| | **Produzir Recomendações** Nessa atividade devem ser produzidas recomendações quanto aos problemas específicos encontrados sugerindo ações alternativas. | Após a análise, deverão ser coletadas as recomendações dos analistas que aplicaram a técnica para adequação desta aos novos projetos de processos. |

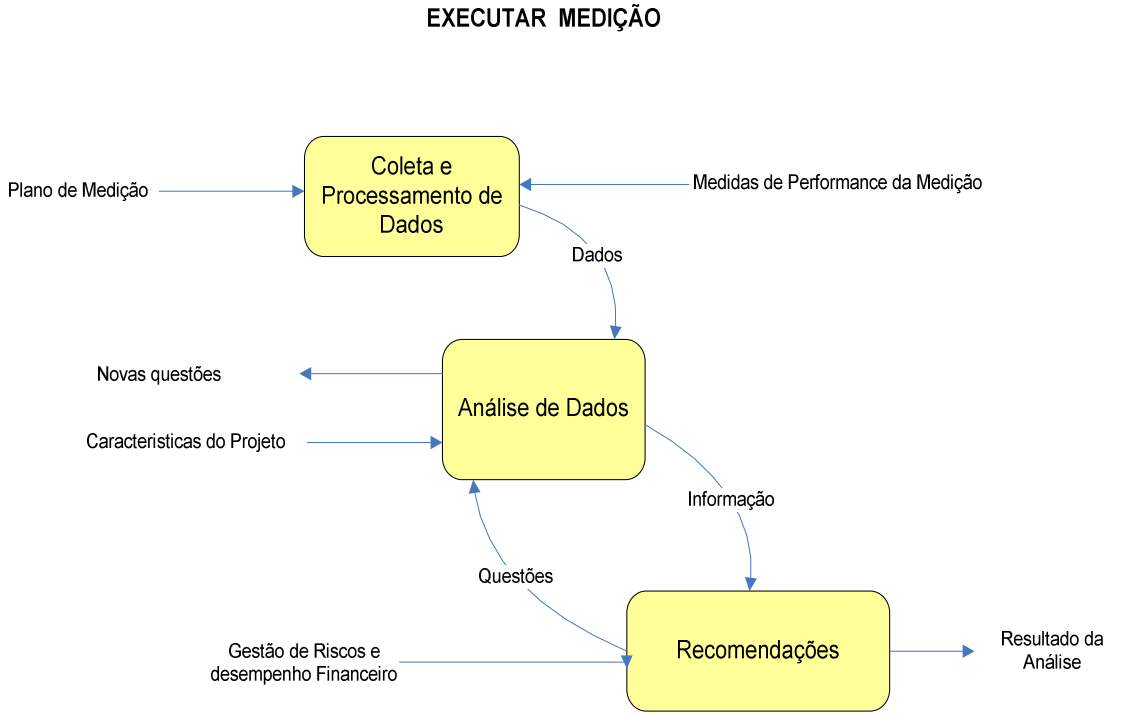



**Figura 21 - Fases da Macro-Atividade Executar Medição - extraído de PSM Measurement Process (PSM GUIDE 4.0B.PART 1.1) adaptado pela autora**

**Tabela 8 - Fase Avaliar a Medição - PSM e sua aplicação no projeto (Fonte: a autora)**

| Fase | Atividades | Adequação do PSM |
|---|---|---|
| Avaliar Medição (Evaluate Measurement): | **Avaliar Medidas** | Essa atividade será realizada juntamente com a fase "Analisar Dados' |
| | **Avaliar o Processo de Mensuração** | Essa atividade será realizada juntamente com a fase "Analisar Dados' |
| | **Atualizar a Base de Experiência** Devem ser armazenados, em um repositório, as lições aprendidas, avaliações, sucessos e fracassos, artefatos e toda a documentação produzida pelo sucesso de mensuração. | Nessa atividade serão coletadas todas as melhorias sugeridas e sua aplicabilidade no estudo proposto. |
| | **Identificar e Implementar Melhorias** Esta atividade objetiva a identificação de alternativas para a melhoria do processo vigente e sua aplicação nos próximos projetos. | De acordo com cada recomendação apontada pelos analistas que aplicaram a técnica para adequação desta a novos projetos de processos, nessa fase serão estudadas como implementar e adequar essas melhorias recomendadas. |

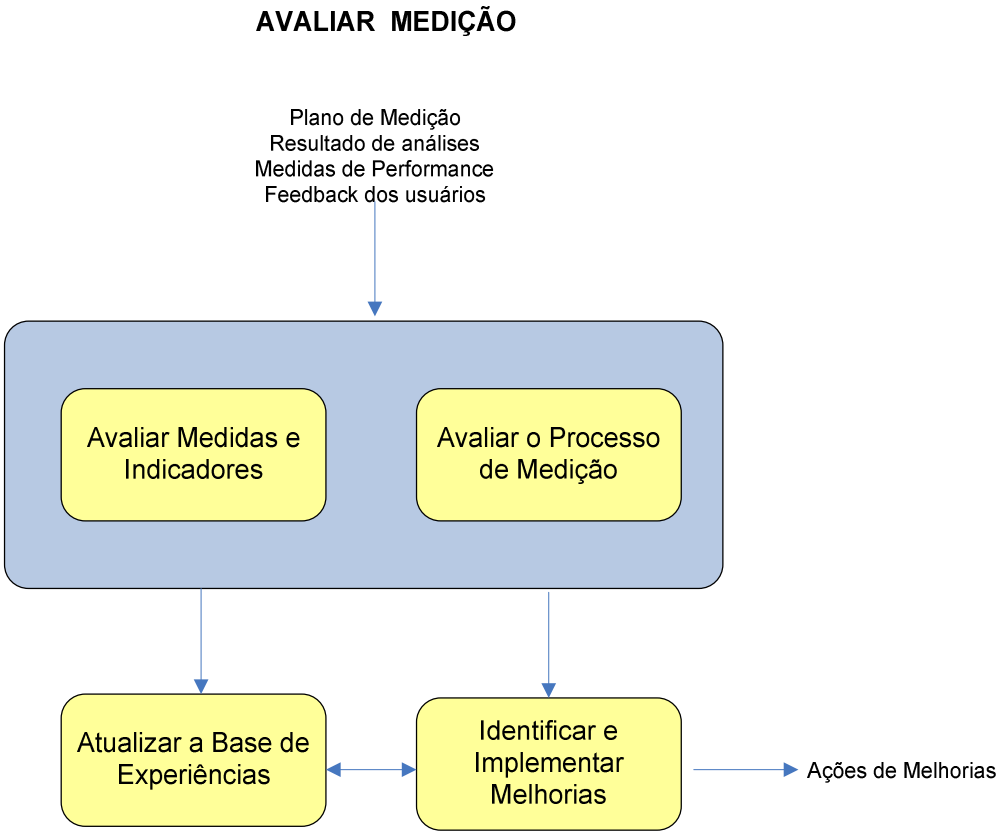



**Figura 22 - Fases da Macro-Atividade Avaliar Medição - extraído de PSM Measurement Process (PSM GUIDE 4.0B.PART 1.1) adaptado pela autora**

A fase "Estabelecer e Sustentar Compromisso" não será aplicada ao experimento.

**Tabela 9 - Fase Estabelecer e Sustentar Compromisso - PSM e sua aplicação no projeto (Fonte: a autora)**

| Fases | Atividades | Adequação do PSM |
|---|---|---|
| Estabelecer e Sustentar Compromisso (Implement Process): | **Obter Comprometimento Organizacional** | Não será aplicada no experimento por se tratar de um estudo. |
| | **Definir Responsabilidades** | Não será aplicada no experimento por se tratar de um estudo. |
| | **Prover Recursos** | Não será aplicada no experimento por se tratar de um estudo. |
| | **Rever o Progresso do Programa de Mensuração** | Não será aplicada no experimento por se tratar de um estudo. |

## 3.2. Cronograma

| ATIVIDADES | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FEV | MAR | ABR | MAI | JUN | JUL | AGO | SET | OUT | NOV | DEZ | JAN | FEV | MAR | ABR | MAI | JUN | JUL | AGO |
| Definição do projeto de pesquisa. | X | X | | | | | | | | | | | | | | | | | |
| Revisão bibliográfica. | | | X | X | X | X | | | | | | | | | | | | | |
| Desenvolvimento inicial da dissertação | | | | | | | X | X | X | | | | | | | | | | |
| Defesa do projeto de dissertação. | | | | | | | | | X | | | | | | | | | | |
| Ajustes solicitados | | | | | | | | | | X | X | | | | | | | | |
| Elaboração do Modelo Conceitual | | | | | | | | | | | X | | X | X | | | | | |
| Aplicação do modelo proposto | | | | | | | | | | | | | | | X | X | | | |
| Validação e Verificação da aplicabilidade do modelo desenvolvido. | | | | | | | | | | | | | | | | | X | | |
| Análise dos resultados obtidos nas duas organizações. | | | | | | | | | | | | | | | | | X | X | |
| Entrega da dissertação | | | | | | | | | | | | | | | | | | | X |

## REFERÊNCIAS BIBLIOGRÁFICAS


ABPMP BPM CBOK®, V 2.0 2010 – **Guide to the Business Process Management** Common Body of Knowledge.

ABREU, Fernando Brito : **Modelo COCOMO: das origens à atualidade**, Interface, nº6, Abril de 1998 a.

AGUIAR, Mauricio. **Estimando os projetos com COCOMO II**. Rio de Janeiro: timetricas. 2004. Disponível em: <http://www.metricas.com.br/Downloads/Estimando_Projetos_COCOMO_II.pdf> Acesso em: 19/08/2012

AGUIAR, M. **Pontos de Função ou Pontos de Caso de Uso? Como estimar projetos orientados a objetos**. Developers' Magazine. V.7, n.77. Jan., 2003.

AGUIAR, Sílvio. **Integração das Ferramentas da Qualidade ao PDCA e ao Programa Seis Sigma**. INDG, 2006.

Aguilar-Savén R. S., **Business process modeling: review and framework**. *International Journal Production Economics*, nº. 90, pp. 129 – 149, 2004.

BARNES, Ralph M. **Estudo de movimentos e de tempo: projeto e medida do trabalho**. São Paulo: Edgard Blücher, 1997.

BARROS, Daniel Barroso. **Governança de Processos: Proposição de um modelo teórico de governança para a gestão de processos.** 2009. 133 f. Dissertação (Mestrado) - Curso de Engenharia de Produção, Departamento de Coppe, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2009.

BOEHM, Barry W.; **Software Engineering Economics**. Prentice Hall, 1981.

BOEHM, Barry et al.; **Software Cost Estimation With COCOMO II**. Prentice Hall, 2000.

Borges, E. P. 2003. **Um Modelo de Medição para Processos de Desenvolvimento de Software**. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Março, 2003.

CAMPOS, V. F. **TQC: gerenciamento da rotina do trabalho do dia-a-dia**. Rio de Janeiro:Bloch, 1994.

CAPPELLI, Claudia et al. **Pesquisa em Estimativas em Projetos de Modelagem de Processos.** 0025. ed. Rio de Janeiro: Unirio, 2009. 52 p.

FURLAN, José Davi (Brasil). Abpmp Internacional. **Guia para o Gerenciamento de Processos de Negócio:** Corpo Comum de Conhecimento. 2.0 São Paulo, 2009. 247 p.

J. Garcia Molina, M. José Ortín, Begona Moros, Joaquin Nicolas, and Ambrosio Toval, **"Towards Use Case and Conceptual Models through Business Modeling**", ER2000 Conference, LNCS 1920, Springer-Verlag Berlin Heidelberg, 2000, pp. 281-294.

KANJAN Thammarak, **"Survey Complexity Metrics for Reusable Business Process"**, ACTIS2010.

KERZNER, H. **Project Management**: A System Approach to Planning, Scheduling, and Controlling - 8th ed. John Wiley & Sons, 2003.

KIVAL, Chaves Weber; ROCHA, Ana Regina Cavalcanti; **Qualidade e Produtividade em Software**. 3ª Edição, Makron Books, 1999.

MCCABE, Thomas J. & Watson, Arthur H. "**Software Complexity**" Crosstalk, Journal of Defense Software Engineering 7, 12 (December 1994): 5-9

MCGARRY, J., Card, D., Jones, C., Layman, B., Clark, E.,Dean, J., & Hall, F. 2002. **Practical Software Measurement: Objective Information for Decision Makers.** Addison-Wesley.

MAFFEO, Bruno. **Engenharia de Software e Especificação de Sistemas.** Rio de Janeiro: Campus, 1992. 516 p.

MARSHALL JUNIOR, Isnard *et al.* **Gestão da Qualidade**. Rio de Janeiro. FGV, 2006.

PILLAI K., SUKUMARAN Nair V.S. **"A model for software development effort and cost estimation"** In: IEEE Transactions on Software Engineering, vol. 23, nº.8, Aug. 1997

PROJECT MANAGEMENT INSTITUTE – PMI. **Guide of Project Management Body of Knowledge** - PMBoK, 2000.

PRESSMAN, Roger S. **Engenharia de software.** 5. ed. São Paulo: Makron Books, 2002. 872p.

PRESSMAN, Roger S. **Engenharia de Software**. 3. ed. São Paulo: Pearson – Makron Books, 1995.

PRESSMAN, Roger S. **Engenharia de Software.** 6. ed. São Paulo: Mcgraw-hill, 2006. 720 p.

PROJECT MANAGEMENT INSTITUTE (PMI), **PMBOK Guide**, 4ª edição, 2008.

PSM. **Practical Software Measurement**, Software Engineering Institute, Carnegie Mellon University, 2012. Available at http://www.psmsc.com.

PUTNAM, Lawrence H.. A General Empirical Solution to the Macro Software Sizing and Estimating Problem. In _____. **Putnam Estimates** IEEE Transactions Software Engineering, V.4 N.4, 1978. p 345 – 361.

RODRIGUES, Marcus Vinicius Carvalho. *Entendo***, aprendendo, desenvolvendo qualidade padrão seis sigma**. Rio de Janeiro, Qualitymark, 2006.

ROVAI, Ricardo Leonardo; SILVA, Marcello Muniz; CAMPANÁRIO, Milton de Abreu. **Metodologias de Estimativa de Prazos,Custos e Orçamentos em Projetos de T.I**. In: 1ºCONTECSI CONGRESSO INTERNACIONAL DE GESTÃO DE TECNOLOGIA E SISTEMAS DE INFORMAÇÃO, 1., 2004, São Paulo. **Metodologias de Estimativa de Prazos,Custos e Orçamentos em Projetos de T.I.** 21-23 de Junho de 2004 Usp/são Paulo/sp - Brasil: Usp, 2004. v. 1, p. 1 - 22.

SOMMERVILLE, Ian. **Engenharia de Software.** 8. ed. São Paulo: Pearson Education - Br, 2007. 568 p.

TUMAN, G. J. **Development and implementation of effective project management information and control systems**. In CLELAND, D. I; KING, W. R. Project management handbook. New York: Van Nostrand Reinhold, 1983.

TRINDADE, André Luiz Presende; PESSÔA, Marcelo Schneck De Paula; SPINOLA, Mauro De Mesquita. **COCOMO II: uma compilação de informações sobre a nova métrica**. In: CONGRESSO INTERNACIONAL DE ENGENHARIA INFORMÁTICA DA UNIVERSIDADE DE BUENOS AIRES,1999, Buenos Aires. **Congresso Internacional de Engenharia Informática.** Buenos Aires: Universidade de Buenos Aires, 1999. p. 1 - 17

VAZQUEZ, C. E., SIMÕES, G. S., ALBERT, R. M., **Análise de Pontos de Função – Medição, Estimativas e Gerenciamento de Projetos de Software**. 3ª. edição. São Paulo: Editora Érica, 2005