CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

EDUARDO JESUS COPPOLA

UMA CONTRIBUICAO À MELHORIA DA MATURIDADE DOS PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE: UM ESTUDO DE CASO DA UTILIZACAO DO *PRATICAL SOFTWARE AND SYSTEMS MEASUREMENT* – PSM INSIGHT

São Paulo
Março/2008

EDUARDO JESUS COPPOLA

UMA CONTRIBUICAO À MELHORIA DA MATURIDADE DOS PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE: UM ESTUDO DE CASO DA UTILIZACAO DO *PRATICAL SOFTWARE AND SYSTEMS MEASUREMENT* – PSM INSIGHT

Dissertação apresentada ao Programa de Pós-graduação do Centro Estadual de Educação Tecnológica Paula Souza, como exigência parcial para a obtenção do Título de Mestre na área de concentração: Inovação Tecnológica e Desenvolvimento Sustentável, sob a Orientação do Prof. Dr. Napoleão Verardi Galegale.

São Paulo
Março/2008

Coppola, Eduardo Jesus

C785u *Uma contribuiçao à melhoria da maturidade dos processos de desenvolvimento de software: um estudo de caso da utilizaçao do pratical software and systems mesaurement – PSM INSIGHT / Eduardo Jesus Coppola. - São Paulo: CEETEPS, 2008.*

127 f.

Dissertação (Mestrado) - Centro Estadual de Educação Tecnológica Paula Souza, 2008.

1. Engenharia de software. 2. Software – desenvolvi mento. I. Título.

CDU 681.3.01

EDUARDO JESUS COPPOLA

UMA CONTRIBUICAO À MELHORIA DA MATURIDADE DOS PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE: UM ESTUDO DE CASO DA UTILIZACAO DO *PRATICAL SOFTWARE AND SYSTEMS MEASUREMENT* – PSM INSIGHT

Dedico este trabalho:

À minha esposa Lilian, por tudo que passamos juntos e pela sua imensa compreensão.

Às minhas filhas Mirella e Ariane, pelo que representam para mim.

Aos meus pais José (*in memorian*) e Cacilda, pelo exemplo.

Ao meu querido amigo irmão Roberto Coppola, pela cumplicidade.

AGRADECIMENTOS

Primeiramente a Deus, por ter iluminado o meu caminho e concedido saúde necessária para o desenvolvimento deste trabalho.

À minha família, pela paciência e compreensão durante o período de estudo e elaboração desta dissertação, após uma fase delicada em minha vida.

À Profª. Drª. Helena Gemignani Peterossi, Coordenadora do Programa de Mestrado e ao Orientador Prof. Dr. Napoleão Verardi Galegale, pelas idéias produzidas durante nossas conversas.

A todos os colegas do curso de mestrado que me acompanharam durante o cumprimento dos créditos.

Por fim, a todos os colegas de trabalho que contribuíram, direta ou indiretamente, para que este estudo se realizasse.

COPPOLA, Eduardo Jesus. **Uma contribuição à melhoria da maturidade dos processos de desenvolvimento de software: um estudo de caso da utilização do *Pratical Software And Systems Measurement* - PSM INSIGHT**, 2008, 125f. Dissertação (Mestrado em Tecnologia) – Centro Estadual de Educação Tecnológica Paula Souza, São Paulo, 2008.

RESUMO

Com a evolução da tecnologia, os projetos de desenvolvimento de softwares se tornaram maiores e mais complexos. Para que seu sucesso seja garantido tornou-se necessário a utilização de planejamentos eficientes. Vários institutos, por meio da engenharia de software, criaram padrões de planejamento para a concepção, desenvolvimento, implantação e manutenção de um projeto, e denominaram essas técnicas de maturidade de software. Para que todo planejamento ocorra é fundamental que existam números, para tanto, surgiram as métricas de software, responsáveis pela quantificação dos processos. Devido ao tamanho e complexidade as métricas também necessitaram de um planejamento de execução, denominado *Practical Software and Systems Measurement* (PSM). Para melhor compreensão desse processo este trabalho foi estruturado em quatro capítulos. No primeiro capítulo são abordados os modelos de maturidade desenvolvidos, com detalhes para os modelos *Capability Maturity Model (CMM) / Capability Maturity Model Integration* (CMMI). O segundo capítulo apresenta as técnicas de métricas de software. O terceiro capítulo demonstra como deve ser planejada a atividade de mensuração. O quarto e último capítulo apresenta um caso prático de aplicação do *Practical Software and Systems Measurement,* numa empresa pública.

Palavras-Chave: maturidade de software, *capability maturity model integration*, níveis de maturidade, métricas de software, *practical software and systems measurement*.

COPPOLA, Eduardo Jesus. **Uma contribuição à melhoria da maturidade dos processos de desenvolvimento de software: um estudo de caso da utilização do *Pratical Software And Systems Measurement* - PSM INSIGHT** , 2008, 125f. Dissertação (Mestrado em Tecnologia) – Centro Estadual de Educação Tecnológica Paula Souza, São Paulo, 2008.

ABSTRACT

With the advance of technology, the development of software projects became largest and more complexes.For its success be guaranteed it started to be necessary the use of efficient planning. Some institutes through the software engineering created standards of planning for the conception, development, implantation and maintenance of a project, and called these techniques as software maturity. For that all planning occurs it is necessary that numbers exist, for this it appeared the metric of software, the responsible ones for the quantification of the processes. Due to the size and complexity the metric also need a planning of execution, called Practical Software and Systems Measurement (PSM). This paper is divided in five parts. At the first chapter the developed models of maturity are boarded. At the second chapter the model Capability Maturity Model (CMM)/Capability Maturity Model Integration is detailed (CMMI). The third chapter presents the techniques of metric of software. The fourth chapter demonstrates as the activity of measuring must be planned. The fifth and last chapter presents a practical case of application of the Practical Software and Systems Measurement.

Key-Words: software maturity, capability maturity model integration, maturity levels, software metrics, practical software and systems measurement

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ACP	Área chave de processo
ACWP	Actual Cost of Work Performed
BCWP	Budgeted Cost of Work Performed
BCWS	Budgeted Cost of Work Scheduled
C4I	Command, Control, Communications, Computers, and Intelligence
CCTA	Central Computer and Telecommunications Agency
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
COCOMO II	Constructive Cost Model
DT&E	Development, Test, and Evaluation
E&MD	Engineering and Manufacturing Development
GAO	General Accounting Office
I-C-M	Issue-Category-Measure (Mapping)
IEEE	Institute of Electrical and Electronics Engineers
IFPUG	International Function Point Users Group
IPECC	Inicialização / Planejamento / Execução / Controle / Fechamento
IPPD	Integrated Product and Process Development
IPT	Integrated Project Team
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
KIA	Conhecimento (Knowledge) / Avaliação (Assessment) / Melhoria(Improvement) – OPM3
LAN	Local Area Network
LOC	Linhas por código
MSP	Managing Successful Programs
OGC	Office of Government Commerce – Antigo CCTA
OPM3	Organizational Project Management Maturity Model
PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
PMMM	Programme Management Maturity Model
PPP	Projeto / Programa / Portifolio – OPM3
Prince2	PRojects IN Controlled Environments
PSM	Practical Software and Systems Measurement
SEI	Software Engineering Institute
SMCI	Inicio (Standardize)/Métricas/Controle/Melhorias (Continuously Improve) – OPM3
T.I	Tecnologia da Informação
WBS	Work Breakdown Structure

SUMÁRIO

INTRODUÇÃO

Com a evolução e o crescimento gradativo da tecnologia da informação, os processos que ela engloba estão deixando de ser uma ferramenta auxiliadora para se tornarem vitais para o negócio. A necessidade da criação de softwares cada vez mais completos, precisos e específicos, exige um melhor planejamento e controle sobre a concepção do software. O impacto das falhas pode ser desastroso para os negócios, pois, os softwares estão se tornando itens chaves para o desenvolvimento da empresa.

Neste cenário, há a necessidade da utilização de metodologias adequadas no desenvolvimento de softwares, aplicando bases quantitativas para a tomada de decisão, garantindo que tais softwares atendam às necessidades atuais e futuras de uma corporação que se visa suprir, reduzindo a incerteza e dando adequada sustentação ao modelo do negócio. Desta maneira, pode-se ressaltar que os principais problemas que afetam os projetos de software não são os tecnológicos e sim os gerenciais. Por isso, tornou-se importante a implantação de modelos da qualidade de software, através de seus atributos, prazos, custos e requisitos atendidos, que promovam a implantação de práticas efetivas para a gestão de projetos e do processo de desenvolvimento de software.

O tema da maturidade dos processos de desenvolvimento de software foi ganhando força na comunidade da engenharia de software, em conseqüência dos resultados práticos obtidos pelas organizações que realizam programas de maturidade com o CMM (*Capability Maturity Model for Software*) como modelo de referência, baseado em algumas premissas, dentre elas: “os maiores problemas nas organizações de software são gerenciais e não técnicos”. Estas premissas apontavam para soluções que, em um primeiro momento, focassem na utilização de princípios básicos de gerência de projeto para “arrumar a casa, gerar resultados imediatos e preparar a organização para as próximas etapas da maturidade”. Sem uma gerência de projetos bem estabelecida, o risco de qualquer outra iniciativa não produzir os resultados esperados é muito grande.

O modelo PSM - *Practical Software and Systems Measurement* surgiu para auxiliar esta gerência na mensuração de projetos de software em relação a custos, prazos e requisitos, dentre eles: horas trabalhadas no projeto, funcionalidades oferecida pelo projeto e ainda permitindo obter o esforço do projeto e seu tamanho funcional através da contagem de pontos de função. O PSM é uma abordagem para melhorar a maturidade dos processos de desenvolvimento de software na sua qualidade e consequentemente dos seus atributos.

O PSM inclui, ainda, um produto de software específico. Trata-se do PSM INSIGHT, uma ferramenta simples que apóia a implantação do PSM em uma organização.

Nas atividades de consultoria em gestão de projetos no desenvolvimento de software, se pôde notar uma grande carência dos profissionais de projetos em relação a modelos específicos de mensuração, bem como de ferramentas que possam auxiliá-los no processo de mensuração robusta de desenvolvimento de software. Talvez a falta de experiência nos modelos e ferramentas de mensuração possa proporcionar alguns fracassos.

Problema

Levando-se em consideração a gestão de projetos, os profissionais que nela atuam e o ambiente que os acolhe é possível compreender que existem, nesse ramo, possibilidades diversas que surgem com os avanços tecnológicos para a melhoria de processos, sendo uma dessas possibilidades o PSM. Daí a motivação para a seguinte questão problematizante: é possível considerar o modelo *Pratical Software and Systems Measurement – PSM* uma *framework* adequada para melhorar a maturidade dos processos de desenvolvimento de software?

Proposta

Colocado o problema e tomando como pressuposto que a área de gestão de projetos de software não é preparada ou não utiliza adequadamente alguns processos de mensuração do desenvolvimento de software, a proposição de uma *framework* baseado no PSM, pode contribuir para o encaminhamento de soluções capazes de resolver o problema de maneira efetiva.

Outro ponto importante é que a forma de estruturação em níveis, de acordo com o CMM (*Capability Maturity Model for Software*) e, fundamentalmente, o CMMI (*Capability Maturity Model Integration*) na organização de um modelo de maturidade vem facilitar a aplicação do PSM e sua compreensão, auxiliando as corporações em relação a sua posição quanto ao processo de maturidade no desenvolvimento de software.

Um modelo da aplicação do PSM INSIGHT, em um projeto de desenvolvimento de software web - IBW (*Internet Banking Web*) pode auxiliar na identificação da situação atual da corporação em relação ao seu nível de maturidade no processo de desenvolvimento de

software para web, ou seja, em que posição esta corporação se encontra em relação às outras com seus padrões internacionais e, se for o caso, qual o nível de maturidade que a organização deseja alcançar. Cabe ressaltar que isto se dá de acordo com seu ambiente e suas próprias características.

Justificativa

O presente estudo se justifica e se torna relevante à medida em que contribui para a produção científica para a área de desenvolvimento de software e TI, aplicadas ao setor de gestão de projetos, uma vez que, atualmente, há pouco conhecimento sistematizado e organizado sobre o assunto. Pela proposta do estudo tratar da utilização e aplicação de um modelo de mensuração do desenvolvimento de software, este poderá ser transposto para outras realidades ou outros setores que tenham características e similaridades abordadas no estudo. Desse modo, este pode se constituir numa referência para novos estudos e para a elaboração de estratégias práticas voltadas a fábrica de softwares, como também ser aplicado em outras áreas, regiões e outras culturas objetivando constatar sua eficácia.

Especialmente para o setor de gestão de projetos de software, a fácil compreensão do modelo de maturidade pode evoluir a cultura organizacional presente no setor, passando de uma TI utilizada apenas como suporte operacional e não aplicada amplamente, para uma estrutura desencadeadora de vantagens competitivas.

Fortalece a justificativa a possibilidade do PSM deve se tornar um elemento padrão para a definição de estratégias de medição nos projetos de software.

Assim, este estudo apresenta a utilização do PSM, de forma tangível e mensurável, num estudo de caso prático visando o desenvolvimento de software para web.

Objetivos

O presente estudo tem como objetivo geral identificar a situação da utilização de um modelo para a mensuração de projetos de software, utilizando o PSM – *Pratical Software And Systems Measurement* e um produto de software específico PSM INSIGHT que pode ser obtido gratuitamente diante de vários modelos de maturidade e propor um modelo de maturidade no desenvolvimento de software web, específicamente, IBW-Internet Banking Web.

Quanto aos objetivos específicos do estudo, almeja-se:

- identificar os principais modelos de maturidade de Projetos e, conseqüentemente, o modelo de maturidade CMM (*capability maturity model for software*), o modelo mais antigo e conhecido, assim como suas tendências junto à literatura nacional e internacional e a integração dos CMMs em compatibilização com a norma ISO, promovendo, assim, o nome *CMM-Integrated*;

- estudar as principais medidas de processos de desenvolvimento de software utilizados na área de gerenciamento de projetos de software, junto à literatura nacional e internacional;

- pesquisar e analisar o modelo PSM *Pratical software and systems measurement*, seu escopo, seu planejamento, sua execução e análise, junto à realidade brasileira, envolvendo uma corporação pública vinculada ao Ministério da Defesa, nas regiões de São Paulo e Rio de Janeiro em suas respectivas capitais;

- propor, com base nos resultados obtidos e nos pressupostos estudados, um modelo de mensuração de processos no desenvolvimento de software, utilizando o PSM como modelo para mensuração e seu respectivo software PSM INSIGHT.

Métodos e Técnicas de Pesquisa

Para a construção do referencial teórico aplicável foi realizada uma pesquisa bibliográfica em fontes primárias e secundárias, com autores nacionais, internacionais e também em periódicos e outras publicações especializadas do setor.

Em outra etapa do estudo foi desenvolvida uma pesquisa de campo "*in loco*" com o departamento de gerenciamento e desenvolvimento de software da corporação no Estado do Rio de Janeiro em sua respectiva capital.

Na pesquisa de campo foram realizadas entrevistas pessoais com os responsáveis pela condução do processo de mensuração, utilizando-se um questionário estruturado, com questões fechadas e algumas abertas, com o objetivo de coletar os dados tais como ocorrem, captar percepções e, principalmente, acompanhar a aplicação e o desenvolvimento do modelo PSM e sua Ferramenta PSM INSIGHT, proporcionando um esclarecimento adequado às questões.

Estrutura do trabalho

Para atingir plenamente os objetivos propostos o presente estudo foi dividido em quatro capítulos, além da presente introdução e da conclusão. Na introdução, é contextualizado o tema do estudo, exposto o problema de pesquisa, hipótese, justificativa, objetivos e metodologia, além da estrutura do trabalho.

No primeiro capítulo são descritos os principais modelos de maturidade, sua evolução ao longo dos anos, destacando sua importância a setores específicos, e o CMM sua evolução, suas áreas de processo e a integração dos CMMs gerando o CMMI.

No segundo capítulo são apresentadas as definições contemporâneas dos processos de medidas de software, além da sua importância na aplicação de medidas de desenvolvimento de software, bem como as principais suas características.

O terceiro capítulo aborda o Modelo PSM- *Pratical Software And Systems Measurement*, com suas respectivas características e aplicações e alguns aspectos da mensuração do processo de desenvolvimento software em relação a seus atributos de qualidade, custos e requisitos.

O quarto capítulo apresenta a pesquisa de campo sobre a utilização do PSM em uma corporação estatal, a metodologia utilizada no trabalho, as técnicas de pesquisa empregadas, as características da amostra escolhida, bem como a demonstração e análise dos resultados, visando descobrir como são aplicadas as técnicas do PSM, como estas estão sendo utilizadas e como contribuem na gestão do gerenciamento do processo de desenvolvimento de software.

Na conclusão, foram feitas as considerações finais, os apontamentos e indicações para trabalhos futuros, para que o modelo possa ser aplicado em outro contexto, outro setor, ou até mesmo servir para a criação de outros modelos, baseados em outras dimensões.

CAPÍTULO I - MATURIDADE DE SOFTWARE

A idéia de maturidade é aparentemente intuitiva, contudo, quando examinada mais longamente, o conceito se revela complexo. Definir um conceito de maturidade para se estabelecer objetivos é, assim, uma tarefa menos trivial do que aparenta a princípio.

Este capítulo enfatiza como a noção de maturidade pode ser relativa e introduz as dificuldades básicas relacionadas ao tratamento desse assunto.

1.1. Definição

Para melhor compreensão das definições de Maturidade e de Software buscou-se no Dicionário Michaelis - Prático da Língua Portuguesa - os respectivos conceitos:

> **ma.tu.ri.da.de** sf (lat maturitate) **1** O mesmo que madureza. **2** Idade madura. **3** Perfeição. M. social, Sociol: grau em que as atitudes, a socialização e a estabilidade afetiva de um indivíduo refletem, como característica normal do homem adulto, um estado de adaptação ou ajustamento ao seu próprio meio (MICHAELIS, 2002).

> **sof.twa.re** sm Inform (ingl) Qualquer programa ou grupo de programas que instrui o hardware sobre a maneira como ele deve executar uma tarefa, inclusive sistemas operacionais, processadores de texto e programas de aplicação. Cf hardware. S. antivírus, Inform: programa que remove um vírus de um arquivo. S. antropomórfico, Inform: V programa antropomórfico. S. beta, Inform: *software* que não foi totalmente testado para comercialização e que, portanto, ainda pode conter erros. S. compatível, Inform: tipo de computador que carrega e executa programas escritos para outro computador. S. de apresentação, Inform: programa aplicativo que permite a um usuário criar uma apresentação de negócios com gráficos, textos e imagens. S. de grupo de trabalho, Inform: aplicação projetada para uso por um grupo de vários usuários, a fim de melhorar a produtividade (p ex, uma agenda eletrônica). S. de compressão em disco, Inform: programa residente que comprime os dados à medida que são escritos em disco, descomprimindo-os quando da leitura. S. de comunicação, Inform: V pacote de comunicação. S. de controle remoto, Inform: programa que funciona em um computador local e um computador remoto, permitindo que um usuário controle o computador remoto. S. de demonstração, Inform: programa que mostra as características de um aplicativo e sua funcionalidade, sem implementar todas as funções. S. de uso intensivo de memória, Inform: programa que usa grandes quantidades de RAM ou espaço em disco durante a sua execução. S. distribuído livremente, Inform: V freeware. S. educacional, Inform: um programa com manuais e vídeo, formando um pacote de treinamento. S. escalável, Inform: aplicação de groupware que permite acomodar facilmente mais usuários na rede, sem a necessidade de investimento em novo *software* (MICHAELIS, 2002).

Relacionando os dois verbetes à área tecnológica, pode-se definir a maturidade de *software como um processo de amadurecimento de software,* ou seja, um conjunto de Modelos e normas, que proporcionam melhores práticas para a melhoria dos processos de tecnologia da informação, dentre eles o desenvolvimento de software.

A utilização dessas normas e regras dá mais credibilidade aos processos, pois os mesmos se tornam gerenciáveis. Previsões de custo e tempo passam a ser mensuráveis e cumpridas dentro dos prazos.

Atualmente existem vários Modelos de maturidade, que são apresentados mais adiante.

1.2. Histórico

A maturidade de software nasceu com a necessidade de melhorar o desempenho nos processos de T.I. A engenharia de software precisava estabelecer padrões de excelência para garantir o sucesso e melhorias nos processos. A partir dessa necessidade, diversos institutos, começaram a desenvolver Modelos de maturidade, dentre eles podemos destacar:

- *Carnegie Mellon® Software Engineering Institute (SEI):* Fundado em 1984 com o patrocínio do exercito americano; é o instituto responsável pela criação dos Modelos *Capability Maturity Model (CMM)* / *Capability Maturity Model Integration (CMMI).*
- *Office of Government Commerce (OGC):* Antigo *Central Computer and Telecommunications Agency (CCTA)*; é o instituto europeu responsável pela criação dos Modelos *PROMPT*, *PRojects IN Controlled Environments (PRINCE2), Managing Successful Programs (MSP).*
- *Project Management Institute (PMI)*: Fundado em 1969, nos EUA; é um Instituto de Gerência de S*oftware*, dentre suas principais publicações esta o *Project Management Body of Knowledge (PMBOK)*, um guia de gerência para Tecnologia da Informação. O instituto também possui um Modelo de maturidade: *Organizational Project Management Maturity Model (OPM3).*

O primeiro Modelo de maturidade denominado *PROMPT* foi desenvolvido em 1975 pela *Simpact Systems*. Esse Modelo foi utilizado pelo governo britânico até 1989 quando foi substituído pelo Modelo PRINCE2, desenvolvido pelo *Central Computer and Telecommunications Agency* (CCTA).

Em 1986 o *Software Engineering Institute (SEI)* iniciou um projeto de desenvolvimento de padrões para a maturidade de *software*. Este projeto desenvolveu em 1993 a primeira publicação do *Capability Maturity Model (CMM).*

Nos anos seguintes, surgiram vários outros Modelos como, por exemplo, o OPM3, o PMMM e o MSP.

1.3. Modelos de Maturidade

Nesse subtítulo são descritos os principais Modelos de maturidade e suas características. Por existir uma espécie de disputa entre os institutos, os Modelos apresentam, basicamente, dois tipos de padrões. O padrão Americano e o padrão Europeu.

No Quadro 01 se pode visualizar os institutos responsáveis por cada Modelo e no Quadro 02 a estrutura e característica de cada um.

Modelo de Maturidade	Instituto Responsável
Projects IN Controlled Environments (PRINCE2)	*Office of Government Commerce (UK)*
Capability Maturity Model (CMM)	*Software Engineering Institute*
Organizational Project Management Maturity Model (OPM3)	*Project Management Institute*
Managing Successful Programs (MSP)	*Office of Government Commerce (UK)*
Programme Management Maturity Model (PMMM)	*ProgM*

Quadro 01 - Modelos de Maturidade
Fonte: Elaborado pelo autor.

Modelo	Estrutura	Característica Principal
PRojects IN Controlled Environments (PRINCE2)	8 Estágios	Justificativa de Escopo
Organizational Project Management Maturity Model (OPM3)	3 Elementos 4 Estágios 4 Dimensões	Busca de soluções já implementadas. Busca de melhoras no processo interno (Portfolio).
Managing Successful Programs (MSP)	6 Fases	Comunicação com a equipe de trabalho. Gerência da equipe de trabalho, atividades e informações.
Programme Management Maturity Model (PMMM)	5 Níveis 9 Áreas de conhecimento	Comparações. Análise de maturidade da organização
Capability Maturity Model (CMM)	5 Niveis	Avaliação de Níveis de maturidade. Otimização dos processos da organização.

Quadro 02 - Características dos Modelos de maturidade
Fonte: Elaborado pelo autor.

1.3.1. *Projects IN Controlled Environments* (PRINCE2)

O modelo de maturidade PRINCE2 nasceu com base nos padrões definidos pelo Modelo PROMPT de 1975. Ele é um Modelo de desenvolvimento de projetos estruturados, criado pelo *Office of Government Commerce*. Suas principais características são:

- Justificativa de escopo: Foco na justificativa do negócio.
- Definição de infra-estrutura e equipe de trabalho.
- Planejamento do produto.
- Divisão do projeto em estágios de gerenciamento e controle.
- Fase de início, intermediária e final controladas e organizadas.
- Revisões regulares do progresso do projeto (*Base-lines*[1)].
- Controle automático de ajustes do projeto.
- Gerência de projeto e *stakeholders*[2] que trabalham em conjunto em todo o projeto.
- Bom canal de comunicação entre a equipe do projeto e o restante da organização.

A estrutura do PRINCE2, é composta por uma *framework* de oito áreas, que juntas tornam o projeto gerenciável e maduro.

A figura 01 demonstra a estrutura do PRINCE2, seguido de um descritivo sobre cada área da *framework*. (OGC,1997).

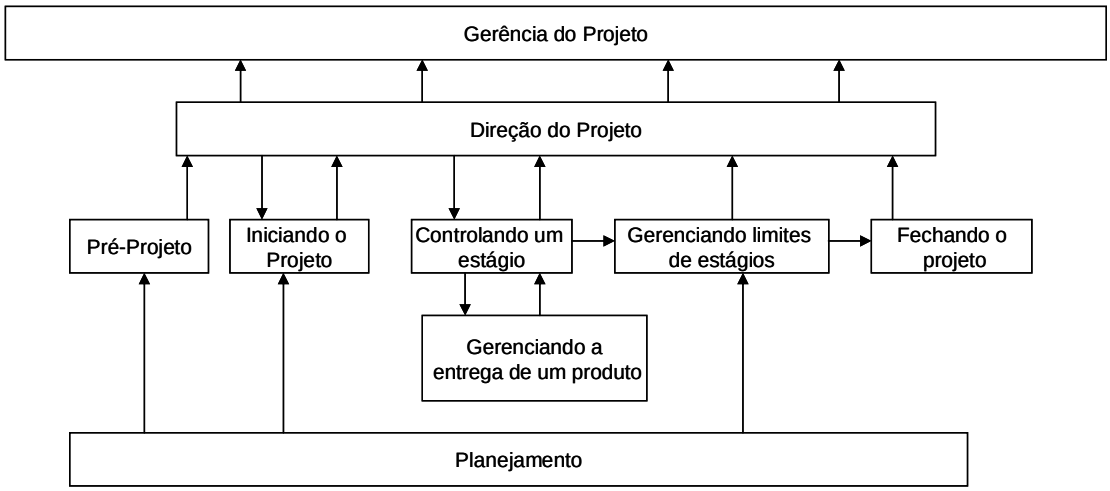


Figura 01 - Estrutura PRINCE2
Fonte: Traduzido de OGC (1997).

[1] *Base- Line*: Linha de Base. Linha de corte.
[2] S*takeholders:* Pessoas influenciadas pelo projeto. (Patrocinadores, clientes, usuários)

• **Direção do Projeto:** Todas as fases do projeto são avaliadas. Essa avaliação ocorre constantemente desde a concepção do projeto até a fase final do mesmo. Ela pode ser feita através de pontos de função, *base-lines*, relatórios e pontos de controle.

• **Pré-Projeto:** Esta é a primeira fase do PRINCE2. Nela é criado um pré-projeto para assegurar que todos os pré-requisitos para o desenvolvimento do projeto estão disponíveis. São definidos três fatores chaves: 1) Permissão de acesso completo às informações necessárias para o desenvolvimento do projeto; 2) Definição da equipe de projeto; 3) Criação de esquema de início de desenvolvimento.

• **Iniciando o projeto:** Com o pré-projeto efetuado, pare-se para a definição mais concreta do que o projeto representa à corporação e quais são seus impactos. Nessa fase têm-se como fatores-chave:

- Justificar a necessidade do projeto.
- Definir uma gerência estável.
- Documentar e validar o Modelo de negócio do projeto.
- Assegurar que exista tecnologia ou uma empresa responsável pelo desenvolvimento do projeto, antes de seu início.
- Garantir que exista recurso para o desenvolvimento do primeiro estágio do projeto;
- Definir *Base-lines*.
- Analisar riscos do projeto.

• **Gerenciando Limites de Estágios** Nesta fase são apresentados resultados do desenvolvimento do projeto à diretoria. Caberá a diretoria definir se o projeto deverá, ou não, ter continuidade.

• **Controlando um estágio:** Esta fase descreve ao gerente de projetos como deverão ser feitas as atividades de monitoração e controle do projeto. O gerente deve assegurar que o estágio de desenvolvimento analisado permaneça em curso e seja ajustável a possíveis erros.

• **Gerenciando a entrega do produto:** O objetivo desta fase é permitir que os produtos do projeto sejam entregues no tempo determinado.

• **Fechando o projeto:** São apresentados à diretoria todos os relatórios de desenvolvimento do projeto, apontando as dificuldades encontradas, alterações feitas no

projeto, cumprimento dos prazos. É validado junto a diretoria a aprovação do projeto e seu encerramento.

• **Planejamento:** O planejamento está presente em todos os processos do PRINCE2. Ele fornece a estrutura de planos de desenvolvimento que, geralmente, seguem uma definição de prioridades, não são utilizadas apenas para um projeto e sim para a empresa.

O objetivo principal do PRINCE2, esta na justificação do negócio. Ele é um Modelo de maturidade que se baseia na justificativa do negocio, no grau de importância que o projeto tem sobre os interesses da empresa. Sua implementação busca melhoras na administração de projetos, controle de riscos, comprimento do orçamento previsto e do tempo (OGC, 1997).

1.3.2. Organizational Project Management Maturity Model (OPM3)

O Modelo de maturidade *Organizational Project Management Maturity Model* (OPM3) foi desenvolvido a partir da necessidade de se estabelecer um padrão de organização para processos de T.I. Durante cinco anos foram acompanhados mais de 800 gerentes de T.I. em mais de 30 países pelo *Project Management Institute* (PMI), que com os resultados obtidos nessa pesquisa foi lançada a primeira versão do OPM3, em dezembro 2003. O principal foco do OPM3 é a busca, em seu portfolio, por projetos e soluções de problemas já estudados. Este Modelo trabalha em conjunto com um Modelo de gerência de projetos denominado PMBOK, desenvolvido também pelo PMI.

A figura 02 apresenta uma definição da estrutura do OPM3 (CHUI, 2005).

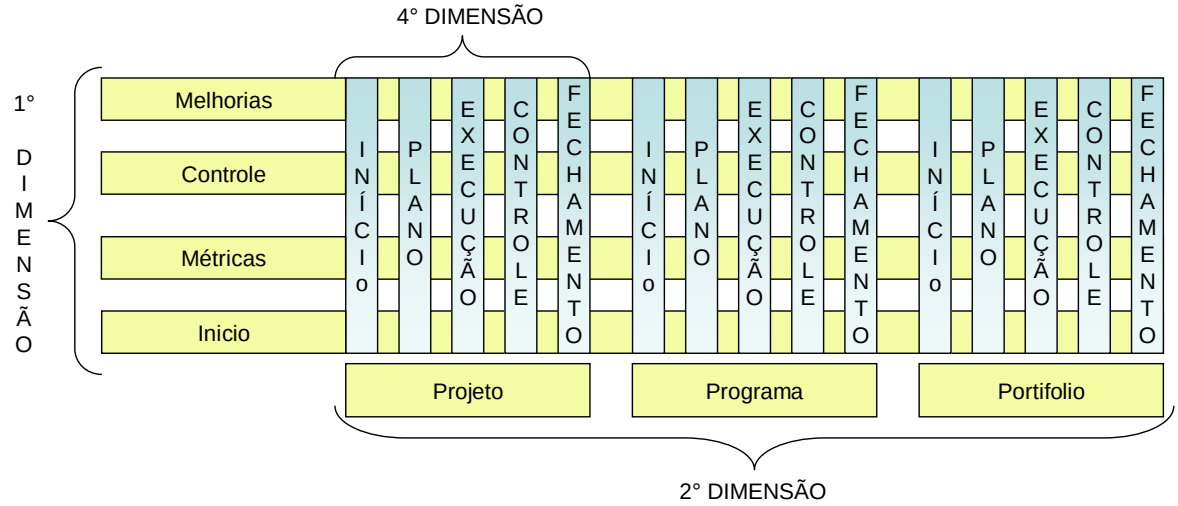


Figura 02 - Estrutura OPM3
Fonte: Elaborado pelo autor.

Examinando a figura tem-se:

1ª Dimensão: Estágios de controle e melhoria. (SMCI – *Standardize, Measure, Control , Continuously Improve*).

- Início (*Standardize*): Levantamento e avaliação de compras, aquisições comuns ao projeto, de acordo com sua necessidade.

- Métricas (*Measure*): Levantamento e aplicação de métricas de performance do projeto e de possíveis riscos.

- Controle (*Control*): Levantamento e controle de desenvolvimento, implementações. Aplicação de auditorias para obter um processo controlado.

- Contínua melhoria (*Continuously Improve*): Identificação de processos com problemas e implementação de melhorias.

2ª Dimensão: Aplicação dos Domínios PPP. (Projeto, Programa, Portfolio).

- Projetos: Criação do produto, serviço. Especificação do projeto.

- Programa: Gerência do projeto coordenada. Obtenção de benefícios.

- Portfolio: Histórico de projetos e programas que poderão facilitar o desenvolvimento deste novo projeto.

3ª Dimensão: O OPM3 não determina a 3° dimensão, fica a critério da equipe de projetos criar essa dimensão de acordo com as necessidades. Geralmente quando é criada a 3° dimensão, ela é a junção da 1 e 2 dimensões.

4ª Dimensão: Implementações do projeto. (IPECC - inicialização, planejamento, execução, controle, fechamento)

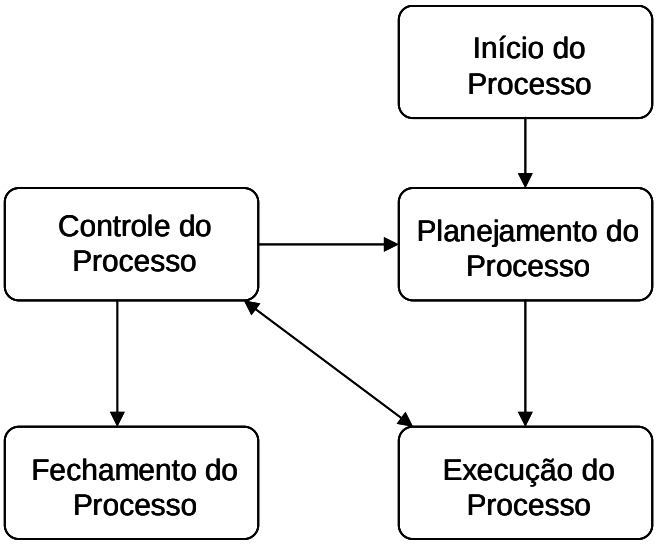


Figura 03 - OPM3 - IPECC
Fonte: Traduzido de OPM3 (2006).

A utilização desta estrutura aumenta a maturidade do projeto gradativamente.

O OPM3 possui ainda 3 elementos-chave que fundamentam as tarefas do projeto para que este se torne maduro. São elas:

- Conhecimento (*Knowledge*): Avaliação do conhecimento da organização sobre o projeto que será realizado. Gerênciamento organizacional do projeto. Fundamentos do OPM3.

- Avaliação (*Assessment*): Avaliação do nível de maturidade atual da organização.

- Melhoria (*Improvement*): Identificação dos pontos críticos da organização e do projeto. Implementação de melhorias nos processos.

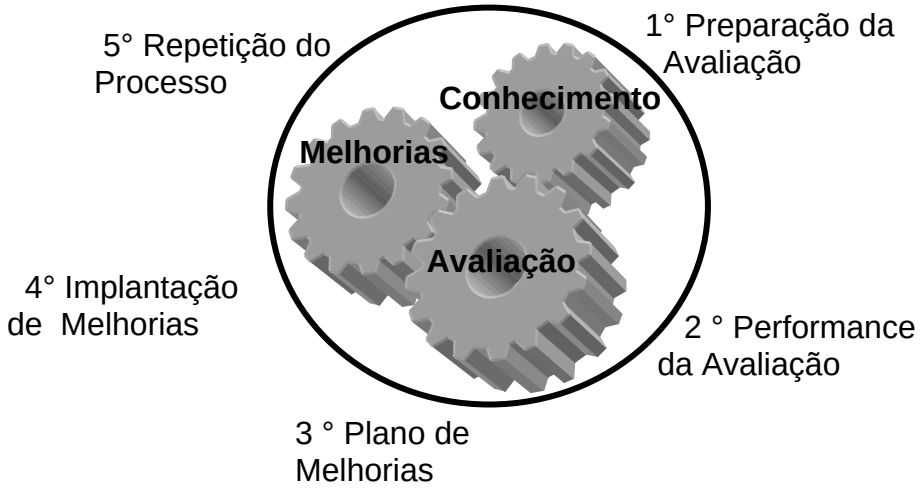


Figura 04 - OPM3 - KIA
Fonte: Traduzido de OPM3 (2006).

O OPM3 é um Modelo de maturidade que busca, em experiências passadas, problemas e soluções já desenvolvidas e solucionadas. Com estas informações é criada uma base de conhecimento (portfolio), que auxilia nas tomadas de decisões do projeto. Em todas as suas fases são implementados os processos de análise, as métricas, o controle e as melhorias. (CHUI, 2005).

1.3.3. *Managing Successful Programs* (MSP)

O *Managing Successful Programs* é uma *framework* de gerência de projetos, criada pelo *Office of Government Commerce*. Seu objetivo é a união de seus três elementos básicos: pessoas, atividades e informações.

O MSP apresenta métodos organizados para alinhar estes três elementos para que a soma deles resulte em um melhor desempenho para o projeto e a empresa.

Suas principais características estão em torno destes seis elementos:

- Organização de pessoas; definindo a responsabilidade de cada um na equipe.
- Prover um ótimo canal de comunicação.
- Planejar as tarefas a serem executadas de maneira que se possam cumprir os prazos.
- Proximidade com os *stakeholders*.
- Efetuar controles de qualidade e de riscos.
- Avaliação de benefícios do projeto para a organização.

Além de ter seu foco na união de seus elementos, o MSP também utiliza um conceito denominado *Vision Statement*, que é um documento que descreve aos clientes externos e internos da empresa, projeções de onde a empresa pode chegar. O *Vision Statement* é uma documentação que informa aos *stakeholders* o objetivo da empresa e de cada projeto, e quais resultados serão obtidos. Com base no *Vision Statement* é criada uma outra documentação denominada *Blueprint*, que é uma descrição detalhada da organização, seus potenciais de negócios, sua infra-estrutura técnica e qualidade de seus serviços.

O MSP possui uma estrutura básica, na qual são descritas as fases que são aplicadas no desenvolvimento do projeto. A figura 05 ilustra a estrutura (OGC, 2006).

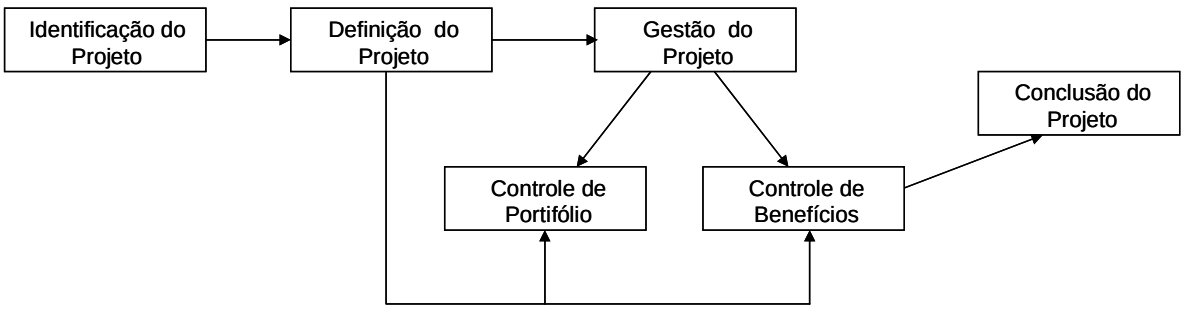


Figura 05 - Estrutura MSP
Fonte: Traduzido de Programmes. About MSP (OGC, 2006).

- **Identificação do projeto:** Nesta fase é criada uma espécie de pré-projeto. O problema é exposto e analisado para verificar se a organização tem capacidade de executar este projeto e também avaliar se ele é realmente necessário.

- **Definição do projeto:** Nesta fase é definida a equipe de projeto a qual desenvolverá, nessa mesma fase, os cronogramas do projeto, a estrutura do projeto, os casos de uso e a definição de patrocinadores.

- **Gestão de Projetos:** Garante que o projeto está sendo executado da maneira correta e dentro dos cronogramas desenvolvidos.

- **Controle de portfolio:** Alinhamento do projeto com os objetivos da organização. Busca de informações para alinhamento do projeto. Monitoração e controle de riscos.

- **Controle de Benefícios:** Monitoria dos benefícios do projeto. Aplicação de métricas para avaliar o desenvolvimento e sucesso ou não do projeto.

- **Conclusão do Projeto:** Revisão e documentação de todo o projeto. Entrega final.

O MSP é um Modelo de maturidade que tem como fundamento estreitar a comunicação entre *stakeholders*, usuários e profissionais de T.I. Todos os processos do projeto, da concepção (*Vision Statement, Blueprint*) à conclusão são documentados e apresentados a todos envolvidos, demonstrando os objetivos, limitações e projeções do mesmo.

1.3.4. *Programme Management Maturity Model* (PMMM)

O *Programme Management Maturity Model* (PMMM) é um Modelo de maturidade desenvolvido pelo ProgM. Esse Modelo foi desenvolvido com base nos cinco níveis do *Capability Maturity Model* e nas nove áreas de conhecimento do OPM3/ PMBOOK. Sua primeira publicação aconteceu em 2002.

Seu objetivo é medir e avaliar os níveis de maturidade de uma organização, propondo as ações que deverão ser tomadas para melhor desenvolvimento e evolução de maturidade.

Para determinar o nível de maturidade da organização, o PMMM possui um questionário por meio do qual são definidos:

- Os comparativos de níveis de maturidade de outras organizações (*benchmarking*);
- A Discriminação de qualidades e erros.
- O desenvolvimento de uma proposta para a melhoria do nível de maturidade.

1.3.5. *Capability Maturity Model* (CMM)

O *Capability Maturity Model* é um modelo de gestão de projetos que proporciona um melhor controle e qualidade no desenvolvimento e manutenção de softwares.

Dois dos principais modelos criados pelo SEI (*Software Engineering Institute*) para melhoria de processos são o SW-CMM e o CMMI. Criado no final da década de 1980 apenas para software, o SW-CMM obteve grande sucesso ao gerar novos padrões como aqueles para a engenharia de sistemas. Posteriormente, como uma evolução dos vários CMMs existentes, foi criado o Modelo CMMI.

"Uma aplicação criteriosa de conceitos de gestão de processos e de melhoria da qualidade no desenvolvimento e manutenção do software" (*Software Engineering Institute*).

1.3.5.1. Definição, Histórico e Subdivisões

O *Capability Maturity Model* é uma metodologia que fornece às empresas regras para controlarem e desenvolverem melhor seus projetos; sua utilização proporciona um aumento de qualidade e controle, alem de prover uma cultura de excelência em engenharia de software.

É composto, basicamente, de cinco níveis de maturidade que levam a empresa a crescer gradualmente nível a nível. O objetivo com o avanço dos níveis é que a empresa tenha autonomia para resolver seus próprios problemas e consiga desenvolver um projeto de software com qualidade, custo e prazos desejados. (PAULK, 1991; WEBER, 1991).

O *Software Engineering Institute,* em 1986, iniciou pesquisas e a construção de um modelo de maturidade, com a funcionalidade de auxiliar empresas e profissionais de T.I. a melhorarem a qualidade e a produção de software (HUMPHREY, 1987). O resultado desse trabalho gerou a primeira versão do *Capability Maturity Model*, em 1993 (PAULK, 1991; WEBER, 1991).

Apesar de o *Software Engineering Institute* ser financiado pelo departamento de defesa dos Estados Unidos (DoD), ele foi o modelo de maturidade que ganhou mais popularidade mundialmente.

Este software criou cinco subdivisões para o *Capability Maturity Model*. Cada uma dessas subdivisões tem a função de avaliar a maturidade de um processo específico, porém, todas possuem as mesmas características básicas (CMU/SEI-TR-010, 2002).

As subdivisões são:

- *Software Engineering Capability Maturity Model*. (SW-CMM): Modelo do Capability Maturity Model atualmente mais conhecido e utilizado. Esse Modelo atua na avaliação da qualidade e controle do software durante seu processo de desenvolvimento.

- *Software Acquisition Capability Maturity Model*. (SA-CMM): Este Modelo avalia os processos de seleção, aquisição e instalação de softwares terceiros na empresa. Sua função é determinar como selecionar o software correto para a atual necessidade da empresa.

- *Systems Engineering Capability Maturity Model*. (SE-CMM): Este Modelo avalia a empresa, em seus processos de organização de engenharia de sistemas. Nesse Modelo, hardware e software, são considerados como um conjunto.

- *Integrated Product Development Capability Maturity Model*.– (IPD-CMM): Este Modelo incorpora ao Modelo SE-CMM, os processos de suporte ao usuário do software.

- *People Capability Maturity Model.* (P-CMM): Este Modelo avalia a maturidade da empresa, nos processos de administração de pessoas. Avalia contratações de mão-de-obra, treinamentos e remuneração da área de T.I.

1.3.5.2. Níveis de Maturidade

A base do *Capability Maturity Model*, como já mencionado anteriormente, é composta por cinco níveis de maturidade; níveis esses que classificam a capacidade (maturidade) da empresa de desenvolver softwares.

Conforme as empresas vão evoluindo nos níveis do *Capability Maturity Model*, sua capacidade de desenvolver softwares se torna mais confiável, atendendo as solicitações do cliente e cumprindo prazos e custos planejados (CMU/SEI-TR-24,1993).

A figura 06 demonstra os cinco níveis de maturidade na sua devida seqüência. Quanto maior for o nível maior será a qualidade de produtividade.

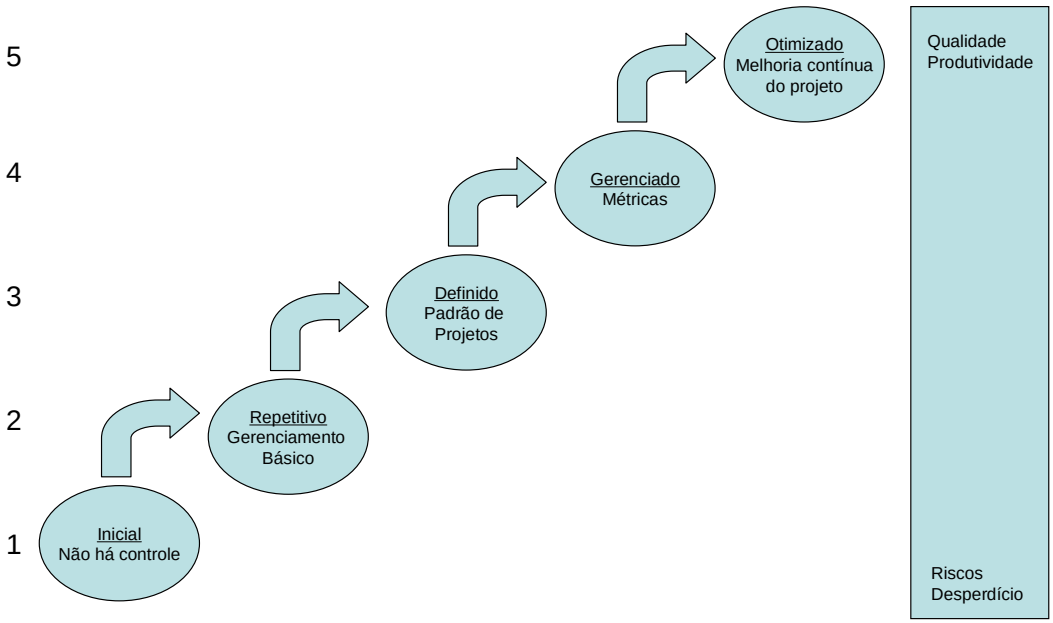


Figura 06 - Níveis CMM
Fonte: ASR Consultoria - Spin-BH (2003)

Nível 1 – Inicial

Nível considerado caótico (ad hoc). A empresa e a equipe de projeto não têm nenhum controle sobre o trabalho, também não existe um ambiente estável para o desenvolvimento e manutenção do software. Os resultados são alcançados pelo talento individual dos membros da equipe.

Uma empresa em nível um, não possui planejamento ou seu planejamento de desenvolvimento é ineficiente, como isso os prazos e objetivos do projeto são quase sempre alterados. Para cumprir as metas, geralmente os projetos deixam de lado o planejamento e partem diretamente para a fase de construção e testes, utilizando o método da tentativa e erro[3]. O sucesso depende do talento individual de cada componente da equipe, contudo, mesmo tendo uma ótima equipe o projeto fica extremamente vulnerável a problemas de gestão.

Neste nível se define que os projetos são imprevisíveis, pois o processo de software é constantemente alterado, assim, também se tornam imprevisíveis de se determinar variáveis como qualidade, tempo e custo (CMU/SEI-TR-010, 2002).

Geralmente é através de planejamento que os projetos se tornam estáveis e não pelo talento individual de uma equipe.

[3] Através dos erros são efetuadas as correções. O *software* é desenvolvido e colocado em teste (tentativa) e os erros que se apresentarem são corrigidos. É um ciclo de tentativas e erros até se chegar ao resultado esperado.

Nível 2 – Repetitivo

Neste nível a empresa já possui um gerenciamento básico de projetos, custos e projeções. As tarefas são definidas e existem cronogramas e prazos a serem cumpridos. O planejamento de novos projetos é baseado na experiência adquirida em projetos anteriores; essa é um das características básicas desse nível, possibilitando a repetição dos processos bem sucedidos e, por este motivo, o nível dois recebe o nome de repetitivo. Outras características desse nível são: documentação de todo o projeto, possibilidade de mensuração e possibilidade de poder ser melhorado.

Os projetos nas empresas de nível dois possuem controles básicos de gestão de software e os gerentes de projeto fazem uso de planejamentos, cronogramas e controle de custos. As alterações durante o projeto são bem administradas e no caso de uma grande alteração são efetuados novos planejamentos, cronogramas e orçamentos. Os itens descritos no planejamento são seguidos fielmente obedecendo prazos e orçamentos (CMU/SEI-TR-010, 2002).

Nível 3 – Definido

Neste nível a empresa possui um padrão de desenvolvimento e manutenção de projetos. Todos os projetos são documentados, integrados e obedecem ao mesmo padrão. Existe na empresa uma equipe de engenharia de software, também conhecida por *Software Engineering Process Group*, responsável pela criação do padrão de desenvolvimento. São criados, também, programas de treinamento para toda a empresa, visando garantir que os gerentes e colaboradores tenham conhecimentos e habilidades suficientes para trabalhar dentro do padrão de desenvolvimento e executar suas funções com a qualidade desejada.

Por meio da padronização dos projetos da empresa, as atividades de gestão e engenharia de software se tornam estáveis e repetíveis. A empresa passa a conhecer melhor suas variáveis de tempo, custo e qualidade.

Nível 4 – Gerenciado

Neste nível todos os processos são quantificados. São coletadas métricas de todos os projetos desenvolvidos; as métricas aplicadas são consistentes e bem definidas com o objetivo de avaliar se os processos estão dentro do padrão de desenvolvimento, bem como do limite aceitável (variáveis de tempo, custo e qualidade).Um banco de dados é criado para armazenar os resultados das métricas de software aplicadas em todos os projetos; este também servirá de base para a análise das métricas.

O nível quatro é considerado um nível previsível, pois todos os processos são mensurados e analisados para verificar se estão ou não dentro dos padrões. Por meio das métricas a empresa determina seus próprios valores de qualidade, tempo e custo de desenvolvimento. Quando uma métrica aponta para um resultado fora do limite, o planejamento permite que o gestor mova ações para a correção do problema. É através da própria métrica que o gestor avaliará o sucesso ou não das suas modificações (CMU/SEI-TR-010, 2002).

Nível 5 – Otimizado

Neste nível toda a empresa está voltada para a melhoria contínua de processos. A empresa tem a capacidade de identificar as melhorias que os projetos e o padrão de desenvolvimento possam sofrer. O objetivo desse nível é prever riscos ou falhas antes que aconteçam, bem como proporcionar uma melhoria contínua do processo de desenvolvimento da empresa.

O nível cinco é considerado um processo de melhoria contínua, pois a empresa sempre estará buscando melhorar seus processos e suas variáveis proporcionando, assim, melhor qualidade, menor tempo e custo.

Nota-se que os níveis são organizados por estágios (nível um ao nível cinco), isso ocorre porque o *Capability Maturity Model* possui uma representação estagiada.

1.3.5.3. Entendendo os níveis

O *Capability Maturity Model* tem como característica do método ser descritivo, normativo e não prescritivo.

Descritivo	Descreve atributos que as empresas devem ter para alcançar as classificações de um nível de maturidade.
Normativo	Define uma norma de como as empresas devem desenvolver projetos.
Não prescritivo	Não informa a empresa como melhorar. O *Capability Maturity Model* informa o que deve melhorar, mas a maneira de como fazer isso caberá a estratégia da empresa.

Quadro 03 - *Capability Maturity Model* – Características do método.
Fonte: Elaborado pelo autor.

Segue o entendimento de cada nível.

Nível Inicial (Nível um)

O nível um é considerado caótico "ad hoc"; mesmo assim, nesse nível, as empresas conseguem desenvolver produtos que funcionam, embora, provavelmente, estejam com o tempo e o orçamento estourados.

O sucesso é abstrato e está totalmente baseado nas competências de cada colaborador. A seleção, contratação e retenção de colaboradores competentes em uma empresa de nível um são fundamentais, pois representam a única chance de um projeto ser concluído.

Nível Repetível (Nível dois)

Tempo de evolução Nível um >> Nível dois: Vários anos

À medida que os projetos crescem em tamanho e complexidade, a necessidade de um plano de organização e gestão é cada vez mais eminente. (SIEGEL, 1990; DoD, 1987).

A organização de um processo de desenvolvimento proporciona aos colaboradores um trabalho mais eficiente, no sentido de incorporar à documentação, as tarefas aprendidas pela equipe de trabalho. Dessa forma, as habilidades necessárias para executar eficientemente os processos são construídas (geralmente, através de treinamento) e melhoradas continuamente, aprendendo-se com as pessoas que estão realizando o trabalho.

Para alcançar o nível dois, a empresa deve implementar um método organizacional e de gestão em seus processos. O objetivo do nível dois é que o desenvolvimento de software seja disciplinado. É necessário que os projetos tenham um planejamento e suas tarefas bem definidas. No nível dois existe um padrão de desenvolvimento para cada projeto, não há um padrão unificado para a empresa toda; este é o próximo passo, ou seja, o nível três de maturidade (CMU/SEI-TR-24,1993; CMU/SEI-TR-010, 2002).

Nível Definido (Nível três)

Tempo de evolução Nível dois >> Nível três: Dois anos

O nível três é a unificação de um padrão de desenvolvimento para toda a empresa. Define-se no nível dois que os projetos necessitam de organização e gestão, porém, esta atividade é feita especialmente para cada projeto. Uma empresa em nível dois pode ter diversas maneiras de gestão e organização.

O nível três cria uma equipe de engenharia de software dentro da empresa, que tem como função unificar todos os processos e desenvolver um padrão de organização e gestão

unificado. Um dos desafios do Nível três é a elaboração de processos que autorizem as pessoas a realizar o trabalho não sendo excessivamente rígidos (HUMPHREY, 1991).

Possuindo um único padrão de desenvolvimento é possível à empresa começar a comparar seus projetos; inicia-se, nesse momento, a análise de variáveis como custo, tempo e qualidade. Esta análise é abordada pelo nível quatro.

Níveis Gerenciado e Otimizado (Nível quatro e cinco)

Tempo de evolução Nível três >> Nível quatro: Dois anos

Tempo de evolução Nível quatro >> Nível cinco: Dois anos

Agora que a empresa já possui um padrão de desenvolvimento, as comparações começam aparecer. O nível quatro é um nível que avalia as variáveis e faz comparações com os demais projetos. Nesse nível é utilizada a técnica das métricas de software, que avaliam como o software está. A empresa agora com a quantificação das variáveis dos projetos inicia o estabelecimento de para suas variáveis.

O nível cinco é a constante otimização dos processos; com os resultados obtidos das métricas, uma equipe avalia e desenvolve novas técnicas visando sempre melhorar os resultados. Neste momento, entende-se que a empresa começa a ser tornar auto-suficiente, pois ela mesma consegue otimizar e corrigir seus processos e cada vez mais seus projetos terão mais qualidade.

Muitas características dos níveis quatro e cinco estão baseadas em um processo estatístico. O diagrama da trilogia Juran ilustra os objetivos básicos da gestão de processos.

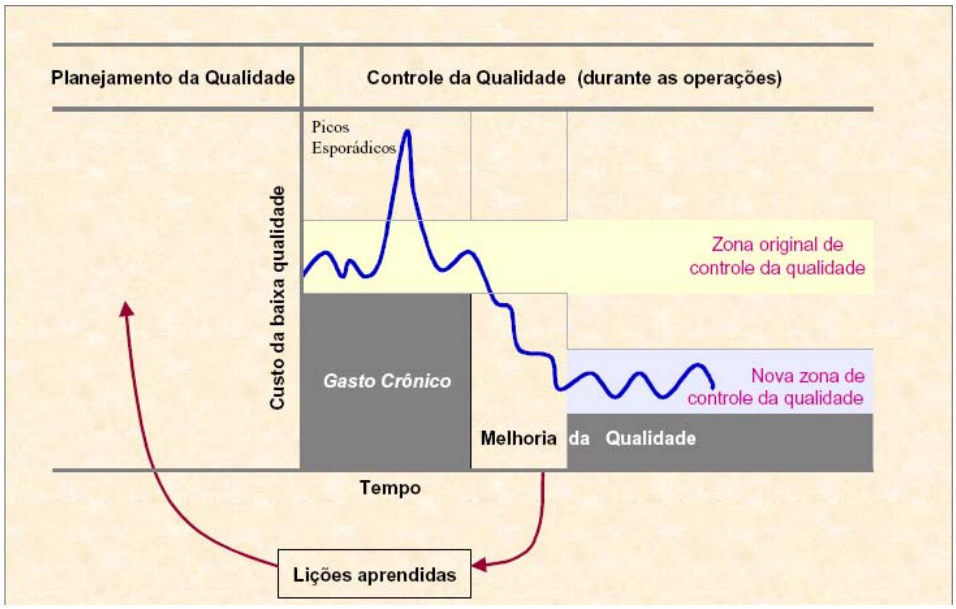


Figura 07 - O Diagrama da trilogia de Juran: Qualidade: Planejamento, Controle e Melhoria
Fonte: Traduzido do CMU/SEI-93-TR-24-CMM V1.1

O foco do nível quatro é o controle de processo. Através das métricas o processo de software é gerenciado criando, assim, zonas de qualidade para a empresa (valores ideais para variáveis como tempo e custo). Quando ocorre algum problema e o processo sai da zona de qualidade, a equipe vai precisar se mobilizar para alinhar novamente o projeto. No gráfico de Juran isto é representado pelos picos esporádicos.

O nível cinco busca a otimização do projeto, com isso são criadas novas zonas de qualidade, representadas por Juran como melhoria da qualidade (CMU/SEI-TR-24,1993).

1.3.5.4. Visibilidade interna dos projetos de software, através dos níveis de maturidade

Os colaboradores envolvidos em um projeto de software possuem uma visão detalhada da situação interna do projeto porque possuem as informações sobre a situação e o desempenho do mesmo. Porém, quanto maior for o projeto, mais difícil se torna a visualização do projeto como um todo; neste caso a visão do projeto é extraída das experiências já vivenciadas e não dos dados reais do projeto.

A visibilidade do projeto se torna gradativamente mais difícil de se avaliar. A visibilidade de um projeto está ligada diretamente à maneira que ele é documentado e planejado.

A Figura 08 demonstra a característica de visibilidade que cada nível da a um projeto (CMU/SEI-TR-24,1993).

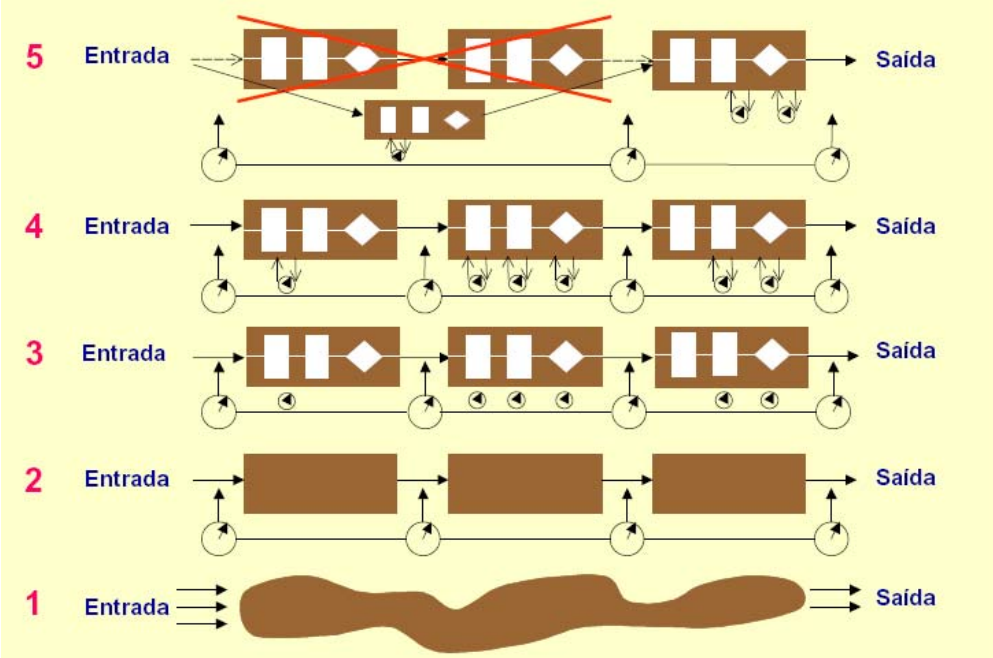


Figura 08 - Visibilidade dos níveis de CMM.
Fonte: Traduzido do CMU/SEI-93-TR-24-CMM V1.1

Nível 1

A visibilidade do projeto é limitada. Como não há uma gestão, não há como prever as próximas tarefas. É muito difícil definir o progresso do projeto. Não há visibilidade ou existe uma visibilidade "ad hoc", pois as tarefas ainda não foram definidas.

Nível 2

A visibilidade do projeto ocorre por pontos-chave e estes determinam uma entrada e saída para cada tarefa; pelo resultado de uma tarefa é possível visualizar o andamento do projeto. Neste nível somente são visualizados esses pontos-chave e os processos de cada módulo não são visualizados.

Os controles de gestão aplicados no nível dois possibilitam que ocorra a visibilidade interna do projeto. Nota-se que o processo de desenvolvimento do software pode ser visualizado como uma série de caixas pretas, permitindo a visibilidade somente nos pontos de transição de uma caixa a outra (pontos-chave). Mesmo que a gerência não conheça os detalhes de cada caixa preta, a visibilidade das transições permite avaliar se o projeto está correto. Neste nível a gerência somente reage a problemas quando eles ocorrem.

Nível 3

A visibilidade do projeto ocorre em todos os módulos. No nível três, por existir um padrão único de desenvolvimento, fica mais fácil a visualização do projeto, pois as tarefas são bem definidas. A estrutura representa a maneira que o padrão de desenvolvimento de software é aplicado aos projetos específicos. Tanto os gerentes como os engenheiros compreendem seus papéis e responsabilidades dentro do processo e entendem como suas atividades se interagem. A gerência se prepara de maneira pró-ativa para os riscos que possam surgir.

Nível 4

A visibilidade do projeto é controlada quantitativamente. Os gerentes são capazes de medir os progressos e os problemas. Eles possuem bases objetivas e quantitativas para tomada de decisões. Suas habilidades para prever resultados crescem constantemente, tornando-se mais precisas as medidas.

Nível 5

A visibilidade do projeto é total. Na busca constante do nível cinco estão os aprimoramentos no desenvolvimento de software. Observa-se isso na Figura 09; do nível um até o quatro o software era desenvolvido como uma estrutura, no nível cinco ele foi otimizado para uma estrutura mais simples.

Os gerentes são capazes de estimar e acompanhar quantitativamente o impacto e a eficiência de uma mudança.

1.3.5.5. Estrutura Interna dos Níveis de Maturidade

Para as empresas se adequarem aos níveis de maturidade, cada nível foi decomposto em partes menores, com exceção do nível 1. As subdivisões dentro dos níveis são: Áreas-chave de processo, características comuns e procedimentos-chave (UNIVERSIDADE FEDERAL DE PERNAMBUCO, 2003).

A Figura 09 demonstra a estrutura interna dos níveis de maturidade.

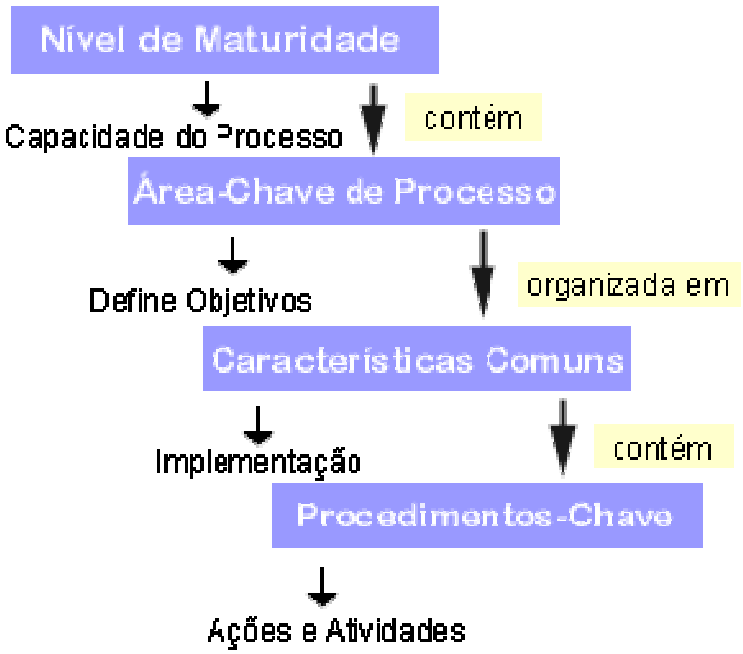


Figura 09 - Estrutura CMM
Fonte: Traduzido do CMU/SEI-93-TR-24-CMM V1.1

1.3.5.5.1. Áreas-chave de Processo – ACP

As áreas-chave de processos são indicadores das áreas na qual a empresa deverá se esforçar para a melhoria do seu processo de software.

Com exceção do nível um, todos os outros níveis de maturidade possuem áreas-chave de processo para serem trabalhadas. Por meio delas uma empresa consegue avaliar qual é o seu nível de maturidade; para que um nível seja alcançado é necessário que todas as áreas-chave do nível desejado estejam em perfeito funcionamento.

Cada área-chave é composta de um grupo de tarefas que deverão ser trabalhadas, estas tarefas são denominadas metas das áreas-chave. As metas resumem as práticas-chave de uma área-chave de processo e podem ser utilizadas para determinar se uma empresa, ou projeto, efetivamente implementou essa área-chave. Ao total são dezoito áreas-chave, distribuídas entre o segundo e quinto nível. Algumas áreas-chave são complementos de outras áreas.

Os quadros que seguem apresentam as áreas-chave do processo de acordo com cada nível.

NÍVEL DOIS As áreas-chaves de processo do nível dois têm o objetivo de implementar controles básicos de gerência e organização.	
ÁREA-CHAVE	**METAS E OBJETIVOS**
Planejamento de Projeto de Software	As estimativas de software são documentadas para o uso no planejamento e no acompanhamento do projeto de software. - As atividades e os compromissos do projeto de software são planejados e documentados. - As equipes e os indivíduos afetados concordam com seus compromissos relacionados com o projeto de software. O objetivo dessa área chave é garantir que o planejamento de projeto seja documentado e que as partes envolvidas (equipe técnica e cliente) estejam cientes de suas obrigações e prazos. O planejamento também deve ser coerente e possível de ser executado. As metas dessa área auxiliam no controle de documentação do projeto e de controle dos prazos.
Acompanhamento e Supervisão de Projeto de Software	Os resultados e o desempenho reais são comparados com os planos de software. - Ações corretivas são realizadas e gerenciadas até a conclusão quando os reais resultados e desempenho desviam significativamente dos planos de software. - As alterações nos compromissos de software são realizadas em consenso pelas equipes e indivíduos afetados. O objetivo dessa área chave é acompanhar o desenvolvimento do projeto. Caso o projeto sofra algum problema, o gerente tem condições de efetuar correções. Quando as correções alterarem os compromissos do planejamento a equipe técnica e o cliente devem estar de acordo com as alterações.
Gestão de Requisitos	Os requisitos de sistema alocados ao software são controlados para estabelecer uma *baseline* para a engenharia de software e para o uso da gerência. - Os planos, produtos e atividade de software são mantidos consistentes com os requisitos de sistemas alocados ao software. O objetivo dessa área chave é estabelecer um entendimento comum entre o cliente e o projeto de software sobre os requisitos do cliente que serão tratados pelo projeto. Este acordo com o cliente é a base para o planejamento e para a gerência do projeto de software. As metas ajudam a determinar a maneira que os requisitos serão alocados para o desenvolvimento do projeto.
Gestão de Subcontratação de Software	O contratante seleciona subcontratados qualificados em software. - O contratante e o subcontratado de software concordam com seus compromissos um para com o outro. - O contratante e o subcontratado de software mantêm comunicações progressivas. - O contratante acompanha os resultados e o desempenho reais do subcontratado, confrontando-os com os seus compromissos. O objetivo dessa área chave é de controlar e gerenciar os contratados de software. Esta área-chave combina os objetivos das áreas-chave: gestão de requisitos, planejamento de projeto de software e acompanhamento e supervisão de projeto de software. Para garantir que os serviços contratados de software estejam funcionando bem as áreas de garantia de qualidade de software e de gestão de configuração de software, também são utilizadas. As metas têm o objetivo de avaliar a qualidade do serviço contratado e de garantir um compromisso com o contratante e contratado.
Garantia de Qualidade de Software	As atividades da garantia da qualidade de software são planejadas. - A aderência dos produtos e das atividades de software aos padrões, aos procedimentos e aos requisitos aplicáveis é verificada. - As equipes e indivíduos afetados são informados sobre as atividades e resultados da garantia da qualidade de software. - As questões de não conformidade que não podem ser resolvidas no projeto de software são encaminhadas para um gerente superior. O objetivo dessa área chave é fornecer uma gestão com visibilidade apropriada sobre os processos utilizados e produtos desenvolvidos pelo projeto de software. Esta área está presente na maioria dos processos de gerência e de desenvolvimento de software.
Gestão de Configuração de Software	As atividades de Gestão de Configuração são planejadas. - Os produtos de trabalho de software selecionados são identificados, controlados e disponibilizados. - As alterações nos produtos de trabalho de software identificados são controladas. - As equipes e indivíduos afetados são informados sobre a situação e os conteúdos das baselines de software. O objetivo dessa área chave é estabelecer e manter a integridade dos produtos do projeto ao longo de todo o ciclo de vida. Esta área está presente na maioria dos processos de gerência e de desenvolvimento de software.

Quadro 04 - Áreas-chave do processo do nível dois

Fonte: Elaborado pelo autor

NÍVEL TRÊS As áreas-chave de processo do nível três têm o objetivo de analisar o projeto e a empresa, já que a empresa possui um modelo padrão de desenvolvimento e uma gerência responsável pelos projetos de software.	
ÁREA-CHAVE	**METAS E OBJETIVOS**
Foco nos Processos da Organização	As atividades de desenvolvimento e melhoria de processo de software são coordenadas em toda a organização. - Os pontos fortes e as oportunidades de melhoria dos processos de software utilizados são identificados com relação a um processo padrão. - As atividades de desenvolvimento e melhoria de processo no nível de organização são planejadas. O objetivo dessa área chave é estabelecer a responsabilidade organizacional pelas atividades de processo de software que melhorem a maturidade dos processos da organização. O primeiro resultado das atividades do foco sobre os processos da organização é um conjunto de recursos de processo de software, descritos em definição do processo organizacional. Estes recursos são utilizados pelo projeto de software como descrito em gestão integrada de software.
Definição do Processo da Organização	Um processo de software padrão para a organização é elaborado e mantido. - As informações, relativas ao uso do processo padrão de software da organização pelos projetos de software, são coletadas, revisadas e disponibilizadas. O objetivo dessa área chave é desenvolver e manter um conjunto utilizável de recursos de processo de software que melhorem o desempenho do processo nos projetos e forneça uma base para benefícios cumulativos e de longo prazo para a empresa. Esses recursos fornecem uma base estável que pode ser institucionalizada por meio de mecanismos tais como treinamento, descrito em programa de treinamento.
Programa de Treinamento	As atividades de treinamento são planejadas. - O treinamento para desenvolver as habilidades e conhecimentos necessários à execução de gerência de software e das funções técnicas é fornecido. - Os indivíduos da equipe de desenvolvimento de software e das equipes relacionadas ao software recebem o treinamento necessário para desempenhar suas funções. O objetivo dessa área chave é desenvolver as habilidades e conhecimentos dos colaboradores, de tal modo que eles possam executar suas tarefas de modo eficaz e eficientemente. Treinamento é uma responsabilidade organizacional, mas o projeto deveria identificar suas necessidades com relação às habilidades e prover o treinamento adequado, quando as necessidades do projeto são específicas.
Gestão Integrada de Software	O processo de software definido para o projeto é uma versão personalizada do processo padrão de software da organização. - O projeto é planejado e gerenciado de acordo com o processo de software definido para o projeto. O objetivo dessa área chave é integrar as atividades de gerência e de desenvolvimento em um processo de software coerente e definido, desenvolvido sob medida, ou seja, adaptado a partir do processo de software padrão da organização e dos recursos de processo relacionados. A Gestão Integrada de Software evolui do Planejamento de Projeto de Software e do Acompanhamento e Supervisão de Projeto de Software do nível dois.
Engenharia de Produto de Software	- As tarefas de desenvolvimento de software são definidas, integradas e realizadas consistentemente para produzir o software. - Os produtos de trabalho de software são mantidos consistentes uns com os outros. O objetivo dessa área chave é executar de maneira consistente um processo de engenharia bem definido que integre todas as atividades de desenvolvimento de software para se obter produtos de software corretos e consistentes de modo eficiente e eficaz. A Engenharia de Produto de Software descreve as atividades técnicas do projeto, tais como, análise de requisitos, projeto, codificação e teste.
Coordenação Intergrupos	- Os requisitos do cliente são divididos entre todas as equipes envolvidas. - Os compromissos entre as equipes de engenharia são divididos entre as equipes afetadas. - As equipes de engenharia identificam, acompanham e resolvem as questões intergrupos. O objetivo dessa área chave é estabelecer um meio para que o grupo de desenvolvimento de software participe ativamente com outros grupos de modo que o projeto esteja mais capacitado a satisfazer as necessidades do cliente. A Coordenação Intergrupos é o aspecto interdisciplinar da gestão integrada de software que se estende para além do desenvolvimento de software; não apenas o processo de software deveria ser integrado, como também as interações do grupo de desenvolvimento com outras equipes deveriam ser coordenadas e controladas.
Revisão por Pares	- As atividades de revisões por pares são planejadas. - Os defeitos nos produtos de trabalho de software são identificados e removidos. O objetivo dessa área chave é remover possíveis problemas dos produtos de software eficientemente e o mais cedo possível. A Revisão por Pares é um método de engenharia importante e eficaz citado na engenharia de produto de software, podendo ser implementado por meio de inspeções e revisões.

Quadro 05 - Áreas-chave do processo do nível três
Fonte: Elaborado pelo autor

NÍVEL QUATRO As áreas-chaves de processo do nível quatro têm o objetivo de quantificar os produtos e processos de software da empresa. As duas áreas-chave de processo neste nível, gestão quantitativa do processo e gestão da qualidade de software, são distintas, ao contrário das outras áreas-chave que são complementares umas com as outras.	
ÁREA-CHAVE	**METAS E OBJETIVOS**
Gestão Quantitativa do Processo	- As atividades de gerência quantitativa de processo são planejadas. - O desempenho do processo de software definido para o projeto é controlado quantitativamente. - A capabilidade do processo padrão de software da organização é conhecido em termos quantitativos. O objetivo dessa área chave é controlar quantitativamente o desempenho do processo do projeto de software. O desempenho de um processo representa os resultados reais obtidos seguindo-se um processo de software. O foco é identificar causas especiais de variação dentro de um processo estável que possa ser medido e corrigido. A Gestão Quantitativa do Processo adiciona um programa de medição abrangente às praticas de: Definição do Processo da Organização, Gestão Integrada de Software, Coordenação Intergrupos e Revisão por Pares.
Gestão da Qualidade de Software	- As atividades de gestão da qualidade de software do projeto são planejadas. - Metas mensuráveis para a qualidade de produto de software e suas prioridades são definidas - O progresso real em direção ao alcance das metas da qualidade para os produtos de software é quantificado e gerenciado. O objetivo dessa área chave é desenvolver um entendimento quantitativo da qualidade dos produtos de software do projeto e obter metas de qualidade específicas. A Gestão da Qualidade de Software aplica um programa de medição abrangente aos produtos de software descritos na Engenharia de Produto de Software.

Quadro 06 - Áreas-chave do processo do nível quatro
Fonte: Desenvolvido pelo autor

NÍVEL CINCO As áreas-chave de processo do nível cinco têm o objetivo de avaliar questões da empresa e do projeto que tratam da melhoria de processo de software de forma contínua e mensurável.	
ÁREA-CHAVE	**METAS E OBJETIVOS**
Prevenção de Defeitos	- As atividades de prevenção de defeito são planejadas. - As causas comuns de defeito são pesquisadas e identificadas. - As causas comuns de defeito são priorizadas e eliminadas sistematicamente. O objetivo dessa área chave é identificar as causas de defeitos e se prevenir contra a sua recorrência. O projeto de software analisa defeitos, identifica suas causas e altera seu processo de software definido, como descrito na Gestão Integrada de Software. As alterações de processo de valor geral são estendidas para outros projetos de software, como descrito na Gestão de Alteração de Processo.
Gestão de Alteração de Tecnologia	- A introdução de mudanças de tecnologia é planejada. - As novas tecnologias são avaliadas para determinar seus efeitos sobre a qualidade e sobre a produtividade.- As novas tecnologias apropriadas são transferidas para a prática em toda a organização. O objetivo dessa área chave é identificar novas tecnologias vantajosas (isto é, ferramentas, métodos e processos) e implementar na empresa de maneira ordenada, como descrito na Gestão de Alteração de Processo. O foco da Gestão de Alteração de Tecnologia é inovar de modo eficiente.
Gestão da Alteração de Processo	- O processo de melhoria contínua é planejado. - A participação nas atividades de melhoria de processo de software ocorre em toda a organização - O processo padrão de software da organização e o processo de software definido para a organização são continuamente melhorados. O objetivo dessa área chave é melhorar continuamente o processo de software utilizado na empresa com a intenção de melhorar a qualidade do software, aumentar a produtividade e reduzir o tempo de desenvolvimento do produto. A Gestão da Alteração de Processo utiliza as melhorias incrementais da Prevenção de Defeitos e as melhorias inovadoras da Gestão de Alteração de Tecnologia e as tornam disponíveis para toda a empresa.

Quadro 07 - Áreas-chave do processo do nível cinco
Fonte: Elaborado pelo autor

1.3.5.5.2. Características Comuns

As características comuns estão presentes em todas as áreas-chave. Elas são atributos que auxiliam a implementação e verificação da tarefa. Essas características proporcionam que a tarefa seja implementada de maneira eficaz e duradoura.

- **Compromissos:** Os compromissos descrevem as ações que a empresa deve tomar para garantir que o processo seja estabelecido e que será duradouro. Geralmente trabalham com políticas empresariais e são definidos pela alta gerência.

- **Habilidades:** As habilidades descrevem as pré-condições que devem existir no projeto ou na empresa para se implementar, de maneira competente, o processo de software. Tipicamente envolvem recursos, estruturas empresariais e treinamento.

- **Atividades:** As atividades descrevem os papéis e os procedimentos necessários para a implementação de uma área-chave de processo. Tipicamente envolvem o estabelecimento de planos e procedimentos, a execução do trabalho, o acompanhamento do mesmo e a tomada de ações corretivas quando necessário.

- **Medições e Análises:** As medições e análises descrevem a necessidade de se medir o processo e analisar as medições.

- **Verificação da implementação:** A verificação da implementação descreve os passos para se garantir que as atividades sejam executadas em conformidade com o processo estabelecido. Tipicamente engloba revisões e auditorias pela gerência e pela garantia da qualidade de software.

As características comuns são como cinco tarefas que todos os processos devem ter, como se dentro de cada processo fossem aplicadas as regras novamente. Todo processo precisa de planejamento, controle, medições e melhorias (CMU/SEI-TR-24, 1993).

1.3.5.5.3. Procedimentos-chave

Procedimentos-chave são atividades que auxiliam que uma meta seja cumprida. Os procedimentos-chave descrevem a maneira e a infra-estrutura que as atividades devem ocorrer para contribuírem nas suas metas.

Cada procedimento-chave consiste de uma sentença simples, freqüentemente seguida por uma descrição mais detalhada, que pode incluir exemplos e explicações mais detalhadas. Esses procedimentos-chave estabelecem as políticas e atividades fundamentais para a área-chave de processo.

Os procedimentos-chave estão listadas em um livro denominado *Key Practices of the Capability Maturity Model*; nas atuais versões do *Capability Maturity Model* os procedimentos-chave estão inclusos na própria versão. Nesta dissertação não estão sendo listados todos os procedimentos-chave.

Nota-se que a união dos procedimentos-chave, com as metas, com as áreas-chave formam o nível de maturidade, portanto, para se obter um nível de maturidade deve-se implementar:

- Os procedimentos-chave para obter as metas.
- As metas para obter as áreas-chave.
- As áreas-chave para obter o nível de maturidade.

1.3.6. *Capability Maturity Model Integration* e seus objetivos

Desde a primeira publicação do *Capability Maturity Model*, em 1993, o *Software Engineering Institute* iniciou um trabalho de coleta de dados para verificar quais mudanças o modelo precisaria sofrer. Os resultados dessas coletas geraram atualizações nas versões do modelo. A figura 10 ilustra as atualizações que ocorreram.

Uma das necessidades de atualização era a integração dos submodelos do *Capability Maturity Model*, que ocorreu em 2000 com a publicação do *Capability Maturity Model Integration*, versão que integrava, em um único modelo, algumas das subdivisões do *Capability Maturity Model*. Esta nova versão conta, também, com mudanças estruturais e de utilização do modelo.

A última atualização que o *Capability Maturity Model Integration* sofreu foi á versão 1.2 publicada em agosto de 2006 (CMU/SEI-TR-008, 2006).

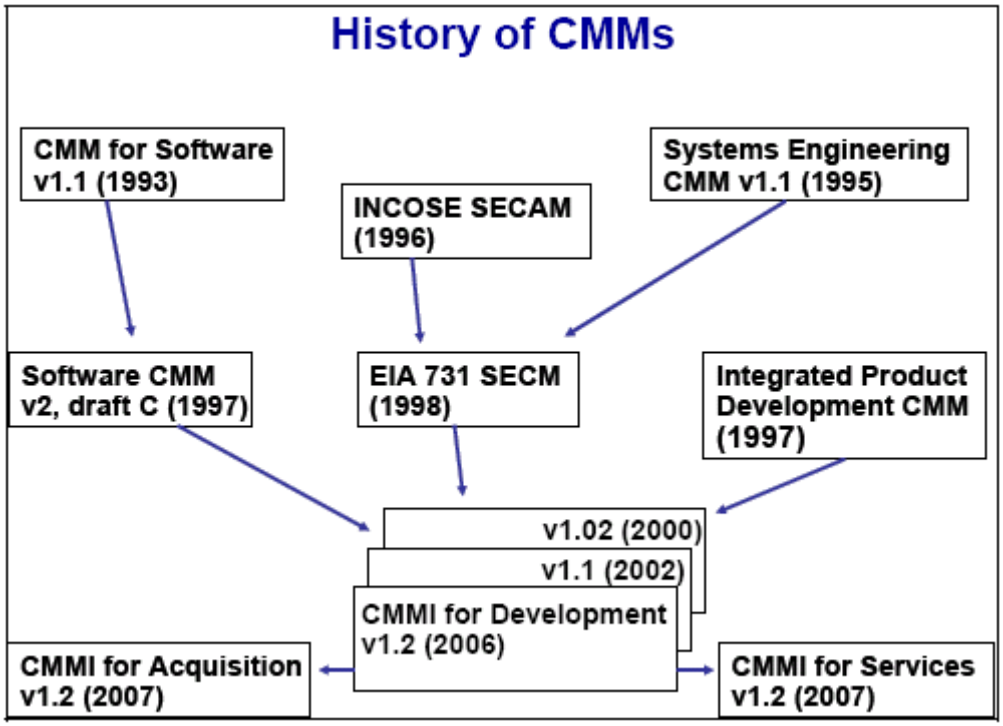


Figura 10 - History of CMMs
Fonte: Guide line CMMI_V1.2 – Agosto 2006

São os objetivos do *Capability Maturity Model Integration*:

- **Compatibilidade à norma ISO 15540[4]:** Norma criada pela SPICE (*Software Process Improvement and Capability Determination*) com o objetivo de criar uma *framework*[5] para avaliação dos processos de *software*. O CMMI atende os requisitos descritos na norma.
- **Integração dos Modelos do *Capability Maturity Model***: A integração dos Modelos do *Capability Maturity Model*, trouxe como vantagem a retirada de itens duplicados entre as versões. Empresas que utilizavam mais de um modelo tinham problemas com compatibilidade entre eles e duplicidade de tarefas.
- **Criação de duas formas de representação - contínua ou estágio**: O *Capability Maturity Model Integration* traz duas versões de representação, uma estagiada herdada do seu antecessor e uma contínua, que permite a empresa trabalhar as áreas-chave de diversos níveis ao mesmo tempo, sem a necessidade de cumprir tarefas de níveis anteriores.

[4] Cf. Para maiores informações acesse: http:// www.iso.org
[5] Um conjunto de objetos que colaboram com o objetivo de cumprir um conjunto de responsabilidades para uma aplicação específica ou um domínio de aplicação,

Os Modelos integrados ao *Capability Maturity Model Integration*, foram divididos em quatro corpos de conhecimento ou disciplinas.

1.3.6.1. Corpos de Conhecimento

As disciplinas Engenharia de Sistemas, Engenharia de Software e Desenvolvimento Integrado de Produto e Processo e finalmente Fontes de Fornecimento, foram desenvolvidas de acordo com as versões evolutivas do *Capability Maturity Model Integration.*

- **Engenharia de Sistemas**

Baseado na versão (SW-CMM) - *Software Engineering Capability Maturity Model*, este corpo de conhecimento é responsável pelo desenvolvimento de sistemas, focando o entendimento das necessidades, expectativas e restrições dos clientes e aplicando isso nos produtos de *software.*

- **Engenharia de *Software***

Baseado na versão (SE-CMM) - *Systems Engineering Capability Maturity Model*, este corpo de conhecimento trabalha as atividades de desenvolvimento de sistema de *software*. A engenharia de *software* objetiva seus esforços na aplicação sistemática, disciplinada e quantitativa para o desenvolvimento e manutenção de *software.*

- **Desenvolvimento Integrado de Produto e Processo**

Baseado na versão (IPD-CMM) - *Integrated Product Development Capability Maturity Model,* este corpo de conhecimento visa disponibilizar uma abordagem sistemática para a obtenção da colaboração necessária das partes envolvidas no ciclo de vida do produto para satisfazer a necessidade e os requisitos dos clientes.

- **Fontes de Fornecimento**

Baseado na versão (SA-CMM) - *Software Acquisition Capability Maturity Model* (SA-CMM), este corpo de conhecimento estabelece melhores práticas para o gerenciamento de aquisição de serviços e produtos de software.

O CMMI integrou quatro modelos do CMM, deixando de fora (P-CMM) - *People Capability Maturity Model*, que foi trabalhado na versão do *CMMI for services*, lançado em 2007.

1.3.6.2. Representação estagiada / contínua

Outra característica do CMMI é a opção de duas formas de representação, uma estagiada (CMM) e uma contínua (ISO-15504).

O objetivo de possuir dois Modelos de representação existe para dar a opção à empresa de escolher o modelo que mais proporcionará resultados sem que haja grandes impactos nos objetivos da organização.

Representação estagiada

A representação estagiada é o modelo utilizado pelos CMM´s; nesse modelo o desenvolvimento da capacitação dentro da empresa ocorre passo a passo. Um nível de maturidade, quando alcançado, garante que a empresa toda esteja trabalhando com aquele nível. Isso ocorre porque nessa representação existe um conjunto pré-definido de Áreas de Processo, onde a organização deve aplicar as melhorias para cada nível de maturidade. Dessa forma, a organização consegue avaliar a qualidade dos seus processos como um todo. (SALES, 2005).

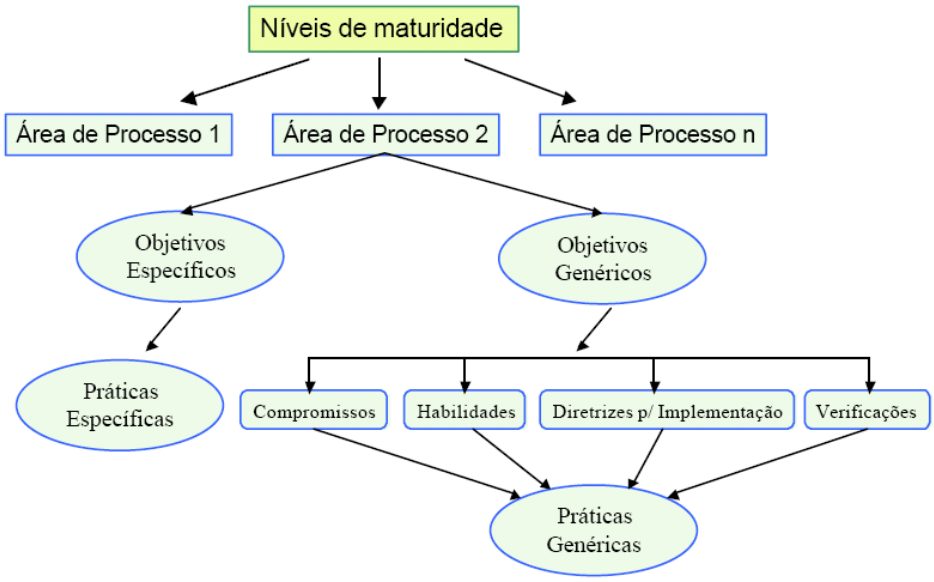


Figura 11 - Estrutura do CMMI na representação estagiada.
Fonte: ASR Consultoria – SinBH.(2003)

As novas características da estrutura do CMMI são abordadas no próximo tópico.

Representação contínua

Representação contínua do Modelo CMMI permite uma empresa aperfeiçoar seus processos para uma determinada área de processo. Nessa representação, a organização pode

focar seus esforços nas áreas que estão mais ligadas aos objetivos da empresa, não existindo a necessidade de atender a todos os processos de uma determinada área de processo para atingir um nível de capacidade.

A estrutura do CMMI para a representação contínua apresenta algumas modificações em relação à representação estagiada, pois não existe mais a necessidade de cumprir todas as áreas-chave de processo para alcançar um nível de capacidade, para a representação contínua, os níveis de capacidade são atingidos à medida que as práticas genéricas e específicas são executadas (SALES, 2005).

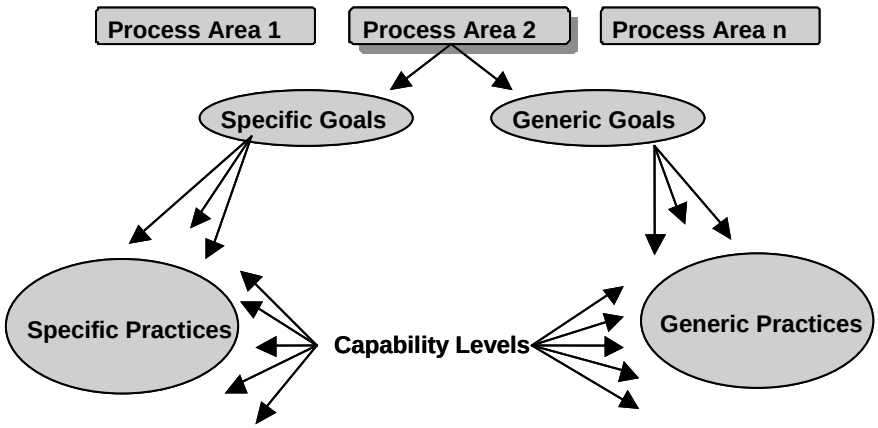


Figura 12 - Estrutura do CMMI na representação contínua.
Fonte: CMU/SEI-TR-008 (2006).

Para utilizar essa representação a empresa pode selecionar uma ou várias áreas de processo. Para cada área de processo existem metas genéricas e específicas. Na Figura 12 pode-se observar bem esta divisão: o topo do fluxo representa a área de processo e imediatamente abaixo as metas determinadas para estas áreas de processo.

Abaixo de cada meta genérica existem práticas genéricas que correspondem a um determinado nível de capacidade, assim como abaixo de cada meta específica existem práticas também específicas que correspondem a um determinado nível de capacidade.

Metas e práticas específicas aplicam-se a uma única área de processo e correspondem ao nível de capacidade de um processo para uma determinada área. (CMU/SEI, 2002).

Metas e práticas genéricas aplicam–se a múltiplas áreas de processo. As metas e práticas genéricas definem uma seqüência de níveis de capacitação que representam melhorias na implementação e eficácia em todos os processos escolhidos para melhoria. (CMU/SEI, 2002).

O objetivo da representação contínua é aplicar as melhores práticas na sua organização, de forma a melhorar processos específicos da organização. (SALES, 2005).

1.3.6.3. Estrutura do *Capability Maturity Model Integration*

A estrutura do CMMI foi desenvolvida a partir da estrutura dos CMM´s. Foram implementadas correções, adaptações e melhorias nos itens que compõem o modelo. A estrutura do CMMI é composta por níveis de maturidade e níveis de capacidade. (CMU/SEI-TR-008, 2006).

Níveis de maturidade

Os níveis de maturidade do CMMI são uma atualização dos níveis do CMM. Alguns dos cinco níveis sofreram alterações e outros permaneceram com a mesma característica, como pode ser observado no Quadro 08.

Inicial	Possui as mesmas características do nível Inicial do CMM.
Gerenciado	Atualização do nível repetitivo do CMM. Foi incorporada a este nível a prática de medir.
Definido	Possui as mesmas características do nível definido do CMM.
Quantitativamente gerenciado	Atualização do nível gerenciado do CMM. As métricas de software, já estão sendo aplicadas desde o nível dois (gerenciado). No nível quantitativamente gerenciado, essas métricas são avaliadas e são definidos objetivos de qualidade.
Otimizado	Possui as mesmas características do nível definido do CMM.

Quadro 08 – Níveis de Maturidade
Fonte: Elaborado pelo autor

Níveis de capacidade

Os níveis de capacidade são utilizados pela representação contínua do CMMI. A principal diferença entre o nível de capacidade e o de maturidade está no cumprimento das práticas genéricas e específicas de uma área de processo. Enquanto que nos níveis de maturidade exigem o cumprimento de todas as áreas-chave para obter a classificação, os níveis de capacidade avaliam individualmente cada área de processo. Este novo conceito

proporciona que a empresa aumente a capacidade das áreas de processos que impactam diretamente em seus objetivos. (SALES, 2005).

O Quadro 09 apresenta os níveis de capacidade:

Zero	Incompleto	A área de processo avaliada, não está sendo executada, ou está sendo executada parcialmente. Uma ou mais metas da área de processo não estão sendo cumpridas.
Um	Executando	Todas as metas da área de processo estão sendo executadas.
Dois	Gerenciado	As metas estão sendo executadas, porém existe um planejamento para a sua execução. Funcionários capacitados controlam o cumprimento das metas e avaliam sua execução.
Três	Definido	O planejamento de execução se torna um padrão para todos os processos da empresa. Com o objetivo de alinhar o desenvolvimento de projetos e contribuir com medições, para que no futuro os processos sejam comparados e possa haver a melhora dos processos da empresa.
Quatro	Quantitativamente Gerenciado	São aplicadas técnicas de métricas de software e estatísticas, para avaliar os processos da empresa. A partir dos resultados são estabelecidos novos objetivos de qualidade.
Cinco	Otimizado	Através das métricas obtidas no nível quatro, o nível cinco busca a otimização dos processos da empresa.

Quadro 09 – Níveis de Capacidade
Fonte: Elaborado pelo autor

Todos os níveis de capacidade são aplicados individualmente por área-chave de processo e seguem um ciclo evolucional do nível 0 ao nível 5. (SALES, 2005).

As áreas-chave de processos[6] foram reformuladas no CMMI. O Quadro 10 demonstra as mudanças que ocorreram com as áreas-chave existentes, e a inclusão de novas áreas. (ASR Consultoria, 2003).

Os quadros que seguem apresentam as alterações ocorridas em algumas áreas-chave, bem como as novas áreas-chave por nível.

NÍVEL DOIS – GERENCIADO **Alterações:** Inclusão de uma nova área-chave: Medição e Análise.	
Área-chave: **Medição e Análise**	**Metas** - Alinhar Atividades de Medição e a Análise. - Prover Resultados da Medição. O objetivo dessa área chave é desenvolver e sustentar uma medida que seja usada para suportar as necessidades de informação da gerência. (CMU/SEI-TR-008, 2006)

[6] No item 1.3.5.5.1. são definidas as áreas-chave de processos e apresentadas as dezoito áreas do CMM.

Quadro 10 – Nível Dois Gerenciado
Fonte: Elaborado pelo autor

NÍVEL TRÊS – DEFINIDO **Alterações:** - Subdivisão da área de processo Gestão integrada de software, em Gerência Integrada de Projeto, Gerência de Risco. - Subdivisão da área de processo Engenharia de produto de software, em Desenvolvimento de Requisitos; - Solução Técnica; - Integração de produto; - Verificação; - Validação; - Área chave de processo Coordenação: intergrupos foi substituída pela Gerência Integrada de Projeto. - Área chave de processo Revisão por pares foi substituída pela Verificação. - Inclusão de uma nova área-chave: - Análise de decisão resolução.	
Área-chave: Gerência Integrada de Projeto	**Metas:** - Usar o Processo Definido do Projeto. - Coordenar e Colaborar com os *Stakeholders*. O objetivo dessa área chave é o desenvolvimento integrado do produto e do processo, a gerência de projeto integrada estabelece uma visão compartilhada para o projeto e uma estrutura para as equipes integradas que realizarão os objetivos do projeto. (SALES, 2005).
Área-chave: Gerência de Risco	**Metas:** - Preparar para Gerenciamento de Risco. - Identificar e Analisar Riscos. - Mitigar (abrandar) Riscos. O objetivo dessa área chave é identificar problemas (riscos) antes que ocorram, para que as atividades possam ser planejadas e não causem impactos para conseguir os objetivos. (CMU/SEI-TR-008, 2006)
Área-chave: Desenvolvimento de Requisitos	**Metas:** - Desenvolver Requerimentos de Cliente. - Desenvolver Requisitos de Produto. - Analisar e Validar Requisitos. O objetivo dessa área chave é analisar o cliente, o produto e as exigências do projeto.
Área-chave: Solução Técnica	**Metas**: - Seleção de soluções de produto. - Desenvolver o design. - Implementar o design do produto. O objetivo dessa área chave é projetar, desenvolver, e executar soluções às exigências do projeto.
Área-chave: Integração de Produto	**Metas:** - Preparar a Integração de Produto. - Assegurar Compatibilidade de Interface. - Unir Componentes de Produto e Entregar o Produto. O objetivo dessa área chave é unir os componentes do produto, garantir que as funções estejam funcionando corretamente e entregar o produto.
Área-chave: Verificação	**Metas:** - Preparar para a Verificação. - Executar Revisões. - Verificar produtos selecionados. O objetivo dessa área chave é assegurar de que os produtos selecionados do trabalho se encontrem com suas exigências especificadas.
Área-chave: Validação	**Metas:** - Preparar para a Validação. - Validar Produto ou Componentes de Produto. O objetivo dessa área chave é demonstrar que um produto ou um componente atenda as especificações quando colocado em seu ambiente real.
Área-chave: Análise de Decisão Resolução	**Metas**: - Avaliar Alternativas. O objetivo dessa área chave é analisar decisões possíveis usando um processo formal da avaliação que busca novas alternativas para atender as especificações estabelecidas.

Quadro 11 – Nível Três Definido

Fonte: Elaborado pelo autor

NÍVEL QUATRO – GERENCIADO QUANTITATIVAMENTE **Alterações:** - As duas áreas-chave de processo, gestão quantitativa de processos e gestão de qualidade de software, transformaram-se nas áreas: Desempenho do processo organizacional e gerência quantitativa do projeto. (SALES, 2005).	
Área-chave: Desempenho do Processo Organizacional	**Metas:** - Estabelecer Desempenho em Linhas de Base e Modelos. O objetivo dessa área chave é estabelecer e manter uma compreensão do desempenho da organização de processos padrão na sustentação de objetivos da qualidade e do processo, e fornecer os dados, as linhas de base, e os Modelos do processo de desempenho para controlar os projetos da organização.
Área-chave: Gerência Quantitativa do Projeto;	**Metas:** - Administrar o Projeto Quantitativamente. - Gerenciamento estatístico de Desempenho de subprocesso. O objetivo dessa área chave é controlar o projeto, para que ele consiga os objetivos estabelecidos com qualidade.

Quadro 12 – Nível Quatro – Gerenciado Quantitativamente
Fonte: Elaborado pelo autor

NÍVEL CINCO – OTIMIZADO **Alterações:** - As duas áreas-chave de processo, Gestão de alteração tecnológica e Gestão de alteração de processos, transformaram-se nas áreas : Análise causal e resolução e Inovação e melhoria Organizacional. - A área-chave Prevenção de defeitos, foi integrada na área chave Análise causal e resolução.	
Área-chave: Análise Causal e Resolução	**Metas:** - Determinar Causas de Defeitos. - Causas de Endereço de Defeitos. O objetivo dessa área chave é identificar causas dos defeitos e dos outros problemas e criar ações para impedir que ocorram no futuro.
Área-chave: Inovação e Melhoria Organizacional	**Metas:** - Selecionar melhorias. - Executar Melhorias. O objetivo dessa área chave é buscar novas tecnologias que poderão ser utilizadas pela empresa, para que seu processo se torne melhor.

Quadro 13 – Nível Cinco – Otimizado
Fonte: Elaborado pelo autor

No capítulo que segue são apresentadas as medidas de software responsáveis pela quantificação dos processos.

CAPÍTULO II - MEDIDAS DE SOFTWARE

As métricas fornecem uma parte importante dos dados necessários para administração de um projeto de software. Este capítulo apresenta características gerais sobre métricas, dificuldades e problemas associados à sua aplicação. Discute, ainda, o uso de informação quantitativa na administração de qualidade de software e desenvolve uma revisão rápida de alguns resultados úteis de estatística, relacionando algumas medidas clássicas.

2.1. A necessidade de medir

Desde os primórdios da humanidade a necessidade de medir acompanhou o homem. A medida é um processo que resulta numa informação objetiva que auxilia o homem a tomar suas decisões, desde controlar transações comerciais até um simples pastor ter o controle de seu rebanho. Com a evolução da humanidade, o processo de medidas passou por várias transformações e evoluções, porém, sua essência contínua a mesma, responder ou quantificar algo para o homem.

Não poderia ser diferente com a área de informática; a necessidade de medir se mostrou presente desde as primeiras soluções tecnológicas, porém, a prática de medir não foi utilizada inicialmente pelas organizações de T.I ou quando se utilizava dessa prática, consideravam-na uma tarefa adicional que não agregava um valor considerável ao projeto. Com o passar dos anos e com os fracassos em projetos, sendo praticamente impossível determinar seu tempo de término e seu custo, as organizações começaram a priorizar as medidas. Nasce, nesse momento, a medida de software, representada por técnicas de medidas para determinar controles e projeções de custos e tempo do projeto. A medida passa a ser um item decisivo para o sucesso do projeto. (PETERS; PEDRYCZ, 2001).

2.2. Medidas de Software

As medidas de software representam uma maneira de mapear um projeto e controlar um sistema tecnológico. Elas fornecem um meio para a quantificação dos processos de um

projeto. Por meio de seus resultados os engenheiros podem tomar melhores decisões para que o projeto seja um sucesso.

As métricas são aplicadas, geralmente, em seis pontos-chave na engenharia de software:

- Custo.
- Testabilidade e manutenibilidade.
- Eficácia.
- Problemas.
- Qualidade.
- Funcionalidade.

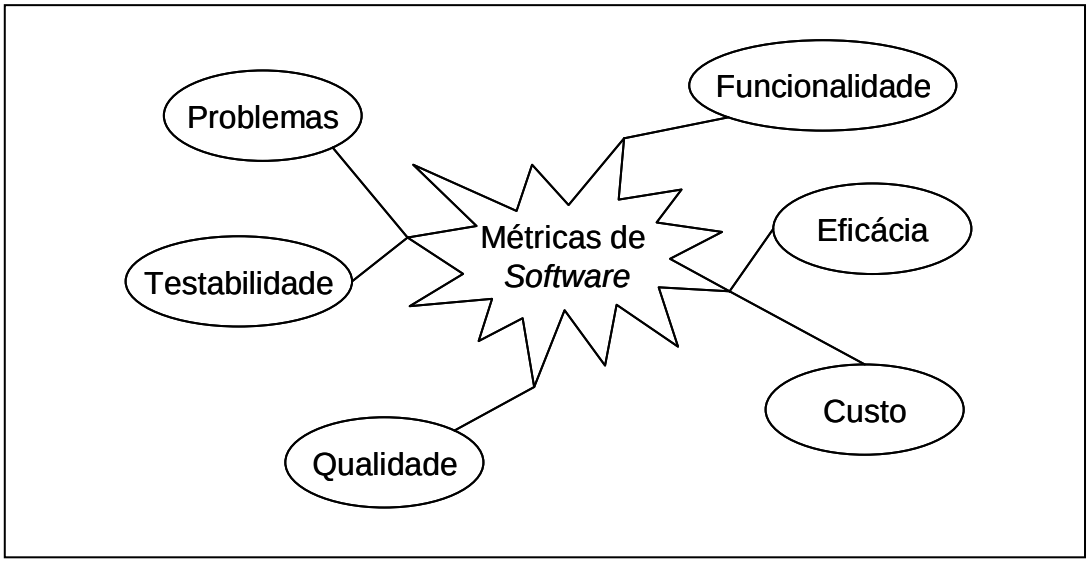


Figura 13 - Áreas que são afetadas pelas medidas de software
Fonte: Adaptado de Gustafson (2002)

As principais métricas de software classificam-se como Medidas de:

- Tamanho.
- Estrutura de dados.
- Estrutura lógica.
- Fluxo de informações.
- Custo.

2.2.1. Medidas de tamanho

Possivelmente são as medidas mais comuns. As medidas de tamanho atuam tanto para o engenheiro de software quanto para o usuário final. Utilizando-as geralmente para avaliar recursos de hardware [7] disponíveis, esforço necessário para a manutenção do sistema e tempo necessário para o desenvolvimento.

2.2.1.1. Medida de linha de código

A medida de linha de código, também chamada por LOC[8], como seu próprio nome diz, conta as linhas de código de um software. Ela é a técnica mais predominante das medidas de tamanho de software.

A utilização desta técnica levanta algumas dúvidas na questão do que contar. Será que devem ser contadas as linhas de comentário do software? Como tratar as funções mais complexas que estão somente em uma linha? Para solucionar essas questões a métrica possui algumas regras de contagem, que são:

- Linhas de comentários e linhas em branco, não serão contabilizadas.
- Todas instruções em uma linha de código são contabilizadas como uma linha, não sendo avaliada a complexidade da operação.

Essa métrica é muito simples de calcular, porém, a principal desvantagem dela é não avaliar a complexidade do código. Ela se torna muito sensível a alterações. Em uma operação escrita de maneiras diferentes (uma otimizada e uma simples) tem-se resultados diferentes.

2.2.1.2.Medida científica de software

Em 1977, Halstead propôs um conjunto de métricas de software conhecido como ciência do software. Dando continuidade no estudo das medidas de tamanho de software, neste modelo são considerados os blocos de instrução de código, avaliando a complexidade de

[7] Equipamento eletrônico utilizado no computador.
[8] Vide apêndice exemplo de aplicação da métrica.

acordo com os recursos utilizados. O problema encontrado na métrica de medida de tamanho, é resolvido aqui, pois é possível fazer comparações de complexidade dentro do mesmo código. (PETERS; PEDRYCZ, 2001).

A ciência de software, parte do ponto que qualquer código de programação é composto de operadores e operandos. Através da contagem[9] desses itens a métrica tem como criar cálculos de: comprimento do programa, medida de volume de software, medida de volume potencial, nível de programa, medidas de esforço e tempo.

Existe uma notação básica na ciência do software, que estabelece que a soma do número de operadores (n1) com o número de operandos (n2) representam o número total (N) de comprimento do código.

São medidas que compõem a métrica da ciência de software:

Comprimento de software.

O comprimento de software pode ser obtido pela simples equação: N = n1 + n2, onde: n1= Operadores e n2= Operandos. Contudo, deve-se prever o comprimento do software baseado nas características da linguagem de programação que será utilizada, e na experiência de desenvolvimento de operações semelhantes. A equação do comprimento estimado é: ^n = n1 log2n1 + n2 log2n2.

Os resultados de comprimento de software real e estimados, quase sempre, não são iguais, pois podem ocorrer impurezas no código e distorções nos resultados. (PETERS; PEDRYCZ, 2001).

Volume de *software*

O volume de software é o número de comparações mentais necessárias para escrever um código de comprimento N. O volume de software é dado pela equação: V = N * log2 n, onde: N é o comprimento do software, log2 n é o vocabulário de codificação. Exemplificando: é a soma dos operandos e operados diferentes utilizados no código. (PETERS; PEDRYCZ, 2001).

Volume Potencial

A medida de volume potencial é a análise mais específica de um bloco do código. O volume potencial é dado pela equação: V* = (2 + n2*) log2(2+n2*), onde: 2 representa o

[9] Cf. anexo contagem de operadores e operandos.

número de entradas e saídas, n2* representa o número de entradas e saídas exclusivas. A principal dificuldade do uso do volume potencial é determinar o número de entradas e saídas exclusivas. Quanto maior for o projeto, mais difícil será para determinar estes valores, portanto, o valor passa a ser não confiável. (PETERS; PEDRYCZ, 2001).

Nível do programa

Os resultados obtidos com o volume potencial são importantes, pois são necessários para efetuar comparações entre os blocos do mesmo código. Esta comparação é denominada como nível de programa, dada pela equação: L = V*/V onde: V representa um volume potencial de um outro trecho de código com o mesmo requisito. Portanto, se V = V*, então L = 1. Se V < V*, então L é menor que 1. O Inverso de L, dado pela equação D = 1/L, é conhecido como uma dificuldade de desenvolvimento.

Pelo motivo de dependência do volume potencial para calcular os níveis de programa, e por ser difícil determinar este valor, Halstead desenvolveu uma nova fórmula para o cálculo de nível, sem a utilização do volume potencial. A equação é dada: ^L = 2n2/n1N2. (PETERS; PEDRYCZ, 2001).

Medidas de Esforço e Tempo

A medida de esforço é calculada a partir do conhecimento do código e das suas estimativas de comprimento, volume e nível. A equação geral do esforço é: E = (N(1)log2 n + n(2)log2n)² / V*

O tempo é calculado pela equação: T = E/ â, onde: E = Esforço e â é o valor de 5 a 20. Um estudo de psicologia determinou que a mente humana pode completar de 5 a 20 ações por segundo. Por este motivo, geralmente são feitos cálculos com a menor possibilidade 5 e com a maior 20, para calcular quanto tempo será necessário para executar um determinado esforço. (desenvolvimento de código). (PETERS; PEDRYCZ, 2001).

Lei de Zipf

A lei de Zipf, 1949, foi desenvolvida para analisar as freqüências que uma palavra aparecia em uma amostra de texto. Verificando esse propósito, era possível adaptar a Lei de Zipf para analisar códigos de software, já que operadores e operandos são repetidos diversas vezes no mesmo código.

A referida lei calcula a freqüência em que ocorre um determinado operador ou operando no código e, pelo cálculo da freqüência de cada item, chega-se ao resultado de comprimento do software. (PETERS; PEDRYCZ, 2001).

2.2.2. Medida de Estrutura de Dados

A medida de estrutura de dados contempla os processos de entrada e saída de dados.

Esta medição considera a estrutura dos dados do sistema, e sua maneira de quantificar se dá somando todas as variáveis do código.

2.2.3. Medida de Estrutura Lógica

A métrica de estrutura lógica analisa a complexidade do código por meio de suas estruturas. O número de estruturas lógicas está diretamente ligado à complexidade do software; quanto mais estruturas existem num único código, mais complexo ele será. Por este motivo foram criadas métricas que contabilizam as estruturas lógicas, dentro de um fluxograma ou código, para avaliar a complexidade do mesmo. (PETERS; PEDRYCZ, 2001).

Complexidade ciclomática do software

Em 1976, McCabe desenvolveu a métrica de complexidade ciclomática[10] do software. Ela quantifica o número de decisões (estruturas lógicas) em um fluxograma ou código.

Sua equação é dada por: v(G) = e – n + 2p, onde: v(G) é o resultado da complexidade, e: Número de caminhos independentes do código, N: Número de nós, 2p: Normalmente p = 1. Portanto, a fórmula é: V(g): e – n + 2.

Foi observado que quanto maior a complexidade de um fluxograma ou código, maior será a complexidade de entendimento do mesmo; com isso tarefas de manutenção, implementação e gerência do código se tornam mais difíceis e mais demoradas. Observando isso, McCabe sugeriu em seus estudos que dez seria um resultado limite razoável para o cálculo de complexidade. Para códigos que ultrapassem esse valor é recomendável que o mesmo seja modularizado[11] obtendo, assim, submódulos (subcódigos) que possuam um valor de complexidade menor.

[10] Vide apêndice exemplo de aplicação
[11] O mesmo que dividir em partes (módulos)

Complexidade ciclomática do *software Método tCTA*

Utilizando os mesmos princípios de McCabe, o método tCTA contabiliza a complexidade do código pelo número de caixas de decisão utilizadas. A equação é dada por: v(G) : DE + 1 , onde : DE é o número de caixas de decisão.

É mais fácil a aplicação do método tCTA, uma vez que este necessita somente da identificação e contabilização das caixas de decisão existentes no código, ao contrário da métrica de complexidade de McCabe, que necessita contabilizar todos os nós e caminhos.

A métrica de complexidade ciclomática é cumulativa. Os resultados obtidos em um módulo do código podem ser somados aos resultados de outro módulo do mesmo código. A equação é dada por: v(G) + v(G).

Medidas de alcançabilidade

Em 1979, Schneidewind e Hoffman criaram a métrica de alcançabilidade[12]. Esta métrica tem o objetivo de avaliar a complexidade de um fluxograma, contabilizando os caminhos exclusivos que cada nó pode alcançar. A equação é dada por: R = número total de caminhos / número total de nós.

O número total de caminhos é obtido pela análise do fluxo ou código. É necessário contabilizar quantos caminhos exclusivos existem para chegar ao nó analisado. (PETERS; PEDRYCZ, 2001).

Medidas de níveis de aninhamento

Em 1980, Dunsmore e Gannon, criaram a métrica de níveis de aninhamento[13]. Seu objetivo é avaliar a profundidade de cada instrução do código. Seu cálculo é feito através da atribuição de níveis de aninhamento para cada instrução de decisão. A atribuição desses níveis deve obedecer as seguintes regras:

- a primeira instrução possui nível de aninhamento 1.
- se a próxima instrução estiver no mesmo nível do que a instrução anterior (não for uma instrução de decisão), o valor de nível de aninhamento permanece.

[12] Vide no apêndice exemplo de aplicação
[13] O mesmo que profundidade

- se a próxima instrução não estiver no mesmo nível (for uma instrução de decisão, *loop*, *if*), o valor de nível de aninhamento é incrementado com o valor 1.

Após atribuídos os níveis , o resultado de aninhamento é obtido através de uma média. (NL = Soma de todos os níveis de aninhamento / número de instruções). (PETERS; PEDRYCZ, 2001).

2.2.4. Medida de Fluxo de Informações

Em 1981, Henry e Kafura introduziram a métrica de complexidade da estrutura do código, concentrando-se no fluxo de informações do módulo. O objetivo é expressar o tamanho e as ligações entre os módulos. A métrica está focada em medir o *fan-in* e o *fan-out*. O *fan-in* é o número de fluxos locais de um módulo, acrescido pelo número de estruturas de dados de que o módulo recupera informações. O *fan-out* é o número de fluxos locais de um módulo, acrescido dos números de estruturas de dados que o módulo atualiza. (PETERS; PEDRYCZ, 2001).

2.2.5. Estimativas de Custos de Software

O desenvolvimento de qualquer produto de software, em geral, é uma tarefa única. Normalmente, os projetos são muito diferentes e as experiências obtidas no passado devem ser utilizadas com muita cautela.

O nível de incerteza de custos é alto. Nem sempre o esforço e o custo gastos em um projeto anterior é um indicador de quanto será o custo de um novo projeto. As estimativas de custos fornecem um valor aproximado do real, cabe ao gerente analisá-lo e decidir entre a melhor. (PETERS; PEDRYCZ, 2001).

Mesmo sendo um valor estimado, estes cálculos são vitais para um projeto. Reifer (1994) definiu seis necessidades básicas para a utilização das estimativas de software.

- Identificar, priorizar e justificar as necessidades do recurso (pessoal, tempo, capital).
- Negociar orçamentos adequados e estabelecer planos de recrutamento pessoal.
- Fazer compensações de custo, produtividade, qualidade.
- Quantificar o impacto dos riscos.

- Avaliar o impacto de mudanças e permitir o replanejamento.
- Modificar orçamentos para atender as contigências (PETERS; PEDRYCZ, 2001)

Atualmente são utilizados seis métodos de estimativas, identificados por Kitchenham (1994 apud PETERS; PEDRYCZ, 2001). São eles:

1. **Opinião dos especialistas:** Avaliação de especialistas com vivência em projetos passados.

2. **Analogia:** Utiliza os resultados obtidos em projetos semelhantes. Este valor é ajustado para o projeto atual. Esse método pode ser muito arriscado, pois o grau de precisão é incerto.

3. **Decomposição:** O projeto é dividido em módulos menores para ser calculada a sua estimativa.

4. **Modelos PERT / Delphi**: Neste modelo o esforço é estimado com base no pior e no melhor possível. As estimativas são combinas utilizando a seguinte fórmula:

$$\text{Esforço} = \frac{\text{Estimativa mais baixa} + 4*\text{estimativa mais provável} + \text{estimativa mais alta}}{6}$$

Baseado neste modelo surgiu o método de Delphi; método que coordena o processo de obtenção de informações e geração de estimativas confiáveis. O método trabalha com as estimativas de vários projetistas, e em cima delas aplica a fórmula de esforço citada acima. É necessário que vários projetistas informem suas estimativas (menor, provável , maior); quanto mais estimativas se têm mais preciso o modelo ficará.

5. **Modelos matemáticos:** Modelos que se baseiam em Matemática e Estatística para estimar os valores. Os modelos mais conhecidos são: Pontos de função de Albrecht, COCOMO, curva de Rayleigh.

Os modelos matemáticos podem ser classificados em:

- Modelos de regressão (paramétricos)

Modelos construídos com base em expressões lineares ou não lineares cujos parâmetros são determinados utilizando análise de regressão e explorando dados experimentais. Exemplo: COCOMO[14], SEER[15].

- Modelos fenomenológicos

Modelos construídos através de percepções gerais. Exemplos: SLIM (Curva de Rayleight)[16].

- Modelos heurísticos

Modelos que incorporam um conhecimento de senso comum e heurística na forma de diretrizes gerais e bem estruturadas. Exemplo: ESTIMACS[17]

- Pontos de Função

Modelo criado por Albrecht (1979); Albrecht e Gaffney (1983); Garmus e Herron (1995). Trabalha com a contagem dos itens de sistema e sua complexidade; seu objetivo é obter uma medida do tamanho do produto que possa estar disponível no início do projeto.

Foi desenvolvido um quadro para classificar os itens de um sistema e sua complexidade; os cálculos de ponto de função são baseados nos resultados obtidos por esta tabela. (PETERS; PEDRYCZ, 2001).

Item	Simples	Médio	Complexo
Entrada externa	3	4	6
Saída externa	4	5	7
Consulta do usuário	3	4	6
Arquivo externo	7	10	15
Arquivo interno	5	7	10

Quadro 14 - Distribuição de variáveis – Modelo Ponto de função.
Fonte: Peters e Pedrycz (2001)

Com base nos resultados obtidos com a tabela de pontos de função são determinadas as seguinte fórmulas:

[14] Maiores informações em: http://sunset.usc.edu/research/COCOMOII Acesso em 20 set 2007
[15] Maiores informações em: http://www.galorath.com/ acesso em 20 set 2007
[16] Maiores informações em: http://www.sc.ehu.es/jiwdocoj/mmis/slim.htm Acesso em 20 set 2007
[17] Maiores informações em: http://www.ca.com/products/estimacs.htm Acesso em 21 set 2007

- Contagem não ajustada de funções: é contabilizado o número de itens e multiplicado pelo seu fator de complexidade. É recomendável resolver a fórmula considerando os três níveis de complexidade.

Exemplo: Duas entradas externas, multiplicado pelo fator simples, resulta em um CNF de seis.

$$CNF = \sum item_i \, p_i$$

- Fator de complexidade técnica: o valor da CNF é multiplicado por um fator. Esse fator é calculado através de uma classificação de 0 a 5 das funções do projeto, onde 0 corresponde a irrelevante e 5 essencial. Este fator é restrito à faixa de 0,65 (todos os fatores irrelevantes), até 1,35 (todos fatores essenciais).

Exemplos de fatores: Funções distribuídas, desempenho, atualização on-line.

$$FCT = 0{,}65 + 0{,}1 \sum fi$$

- Pontos de Função: o valor dos pontos de função é obtido pela multiplicação do CNF com o FCT.

PF=CNF*FCT

O capítulo que segue apresenta um modelo para implementação de medida, o *Pratical Software and Systems Measurement* – PSM, considerando todas as suas características.

CAPÍTULO III - *PRATICAL SOFTWARE AND SYSTEMS MEASUREMENT – PSM*

Poucos são os militantes da área de Qualidade de Software que conhecem o PSM – *Pratical Software and Systems Measurement*, um modelo para a mensuração de projetos de software criado em 1994, também sob o patrocínio do DoD. Perceber a relevância de um processo de mensuração robusto requer experiência (e talvez alguns fracassos) nas atividades de melhoria do processo de software. Neste capítulo, apresentamos o modelo em linhas gerais.

3.1. Definição e Histórico

O *Practical Software and Systems Measurement (PSM)* é um modelo que define a maneira de implementar um processo de medida efetivo. (US ARMY, 2003; AGUIAR, 2002).

O PSM foi criado em 1994 pelo exército dos Estados Unidos, tendo sua primeira publicação em 1997. Ocorreram atualizações no modelo em função da KPA Medição e análise do CMMI, gerando novas versões, sendo que a versão atual é a 4.0c, publicada em 2003.

Hoje, várias empresas participam do desenvolvimento do PSM, sendo elas: Força Aérea, Exército e Marinha Norte-Americanos; *Software Productivity Consortium; Boeing; GTE; Raytheon-Hughes; Lockheed-Martin; MITRE; Software Engineering Institute; Teraquest; TRW.*

Em 2002 o PSM foi formalizado como uma norma ISO/IEC 15939 - *Software Engineering – Software Measurement Process Framework*. (US ARMY, 2003).

3.2. Processo de medida – Escopo do PSM

Cada projeto possui suas características de negócio e técnicas; as métricas devem considerar estes fatores para avaliar corretamente assuntos como: objetivos, riscos, problemas e incertezas que podem ocorrer, bem como aquilo que puder ser feito para que o projeto alcance seu sucesso.

A figura 14 define o escopo do PSM, composto de quatro atividades-chave.

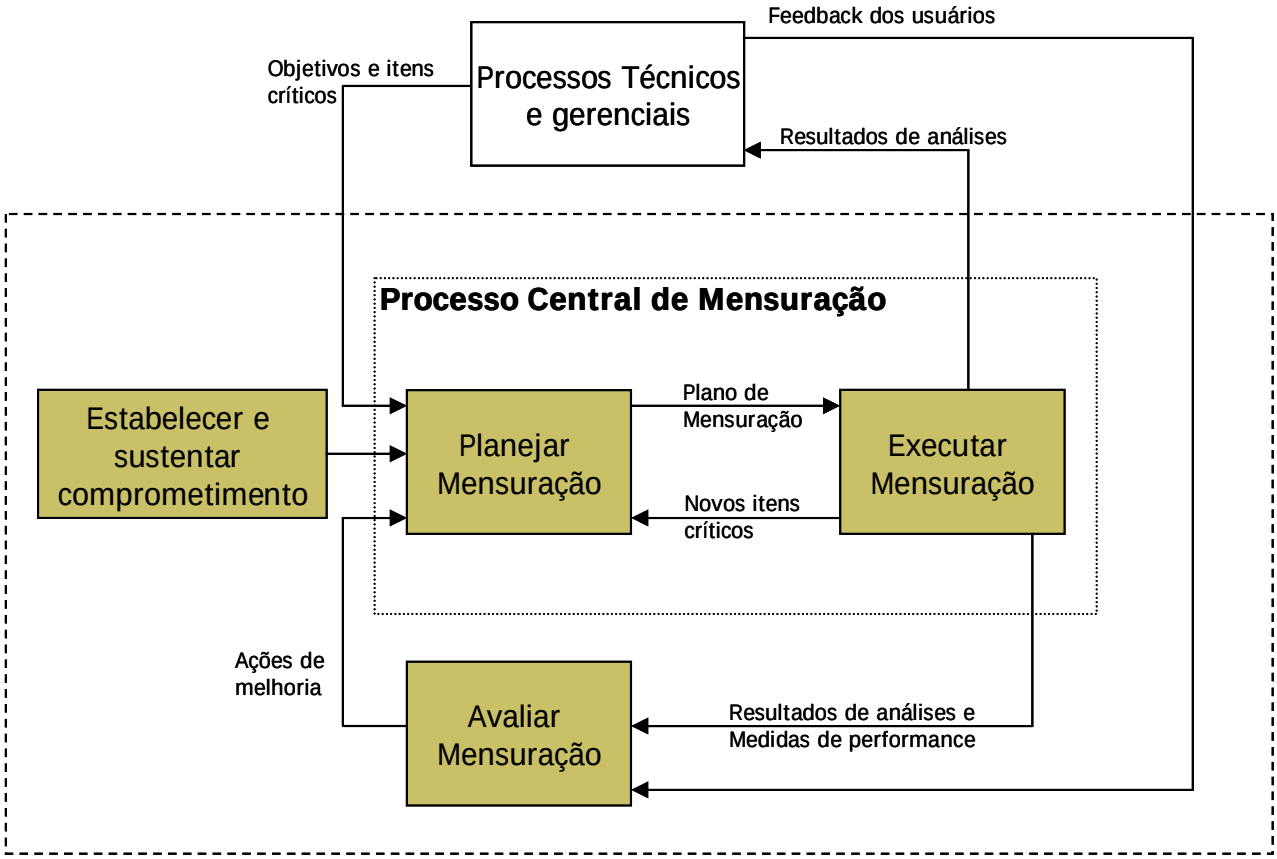


Figura 14 - Escopo do PSM
Fonte: Traduzido do PSM - US. ARMY (2003)

3.2.1. Estabelecer e sustentar comprometimento

Implementar métricas de software representa mudanças significantes na maneira de administração do negócio. Para sustentar essas mudanças, três tarefas-chave atuam como apoio de sustentação do comprometimento com os novos procedimentos.

Conforme a figura 15, as tarefas-chave estão assim representadas:

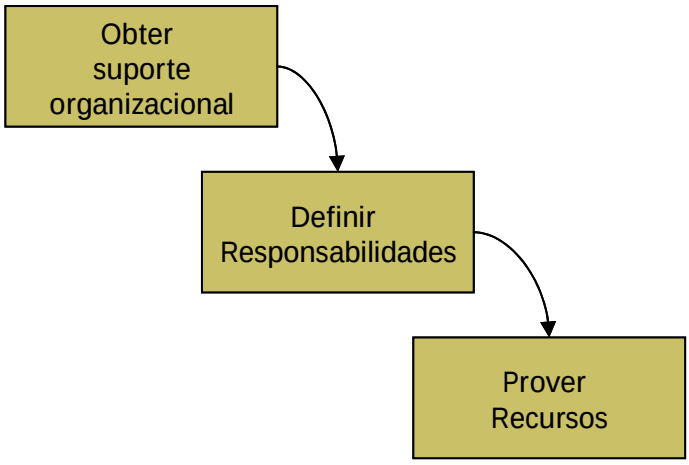


Figura 15 - Estabelecer e sustentar comprometimento
Fonte: Traduzido do PSM - US. ARMY (2003)

• Obter suporte organizacional: Os executivos da empresa devem estar de acordo com o programa de métricas. Eles devem entender como as métricas irão beneficiar seus negócios e apoiar o processo de métricas.

• Definir responsabilidades: Cada colaborador da empresa deve ter sua responsabilidade bem definida quanto ao processo de métricas. Geralmente os colaboradores envolvidos são:

- Gerente Executivo: é o *Stakeholder* do projeto. Através dos resultados das métricas ele tomará as decisões para os negócios da empresa.
- Gerente de Projeto: Responsável pelo projeto. Suas tarefas incluem a análise de métricas e a tomada de decisões.
- Analista de métricas: Responsável pelo plano de métricas e divulgação dos resultados para a empresa.
- Time de Projeto: Equipe responsável pelo desenvolvimento e suporte do projeto. Esta equipe colhe as métricas e as utiliza para efetuar alterações no processo de engenharia do projeto.

• Prover recursos: É a aquisição de ferramentas necessárias e profissionais para implementar as métricas na empresa. (US ARMY, 2003).

3.2.2. Planejar mensuração

A fase de planejar a mensuração tem o objetivo de definir as métricas integrando as partes administrativas do projeto com as partes técnicas, sempre buscando o mais baixo custo.

A figura 16 demonstra o fluxo das tarefas necessárias para planejar a mensuração.

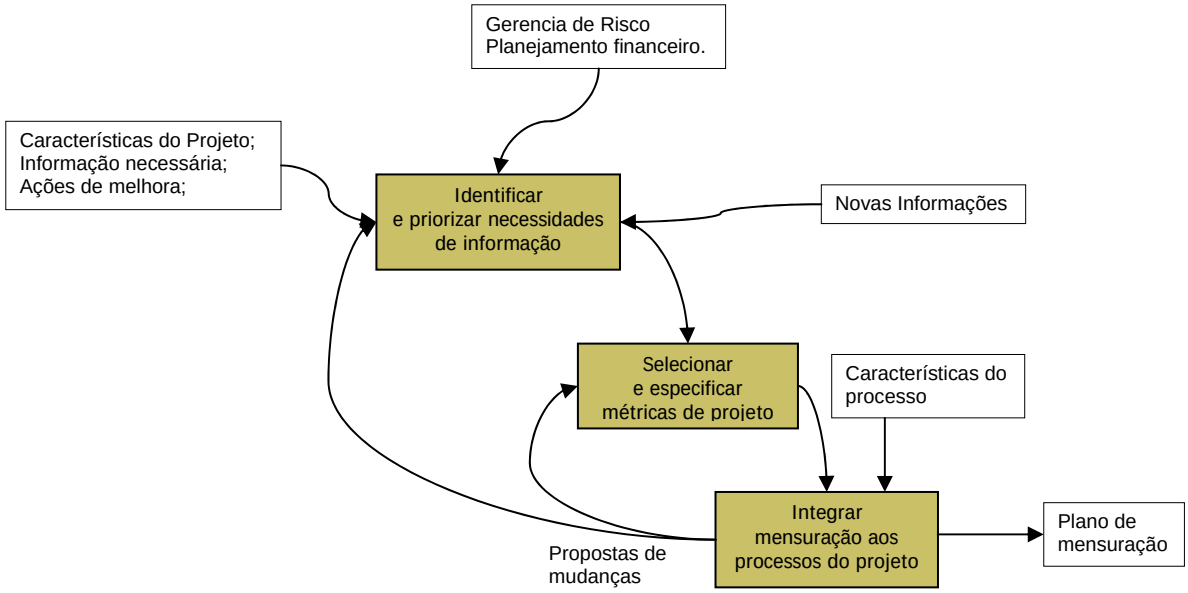


Figura 16 - As atividades do planejamento de mensuração
Fonte: Traduzido do PSM - US. ARMY (2003)

3.2.2.1. Identificar e priorizar necessidades da informação

O processo de medida ajuda o gerente de projeto identificar e administrar problemas ou riscos que podem causar danos ao projeto. O PSM classifica esses problemas como necessidades do projeto. Inicialmente são identificadas as necessidades potenciais de cada área e depois do projeto como um todo. Com as necessidades identificadas é traçado um plano de métricas apropriadas e, finalmente, são priorizadas as necessidades que oferecem um risco maior ao projeto. (US ARMY, 2003).

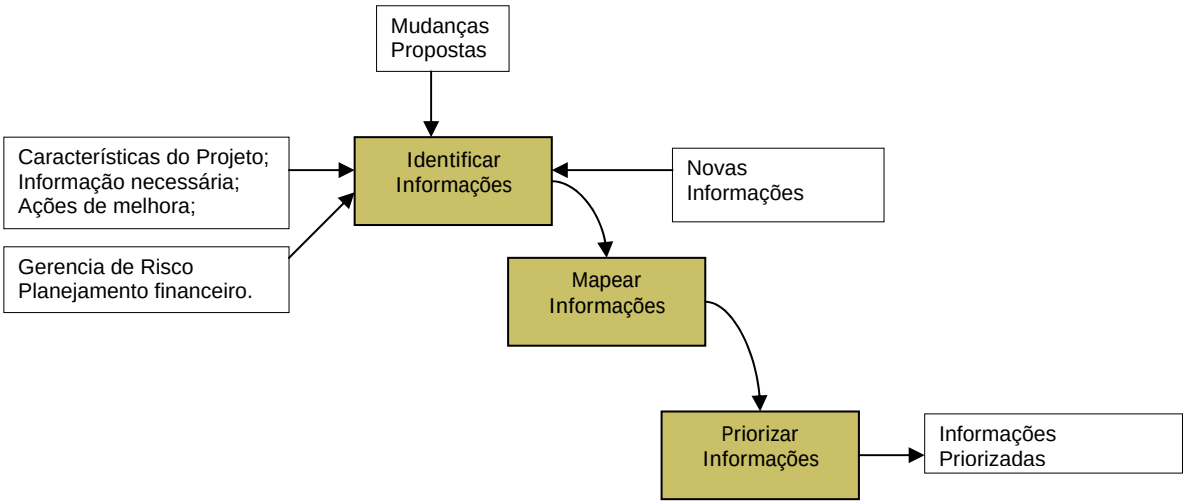


Figura 17 - Identificar e priorizar necessidades da informação
Fonte: Traduzido do PSM - US. ARMY (2003)

- **Identificar Informações**

As primeiras informações de um projeto são seus objetivos. Através deles são definidos o orçamento disponível, o tempo de desenvolvimento e o resultado que é desejado.

Além destas é preciso avaliar as informações que indicam possíveis problemas que o projeto possa sofrer. Para identificá-las devemos avaliar os seguintes itens:

- Avaliação de Riscos: É necessário que o projeto possua uma administração de seus riscos[18]. Eles são fontes de informações de problemas e suas soluções.

- Restrições e projeções de projeto: O objetivo do projeto geralmente está ligado à projeção de desempenho de um cliente. É importante avaliar o mercado que o projeto encontrará quando estiver pronto. Tarefas como esforço, orçamento e qualidade do projeto devem ser avaliadas diante ao mercado futuro.

- Tecnologias de desenvolvimento (*Leveraged Technologies*)**:** O sucesso de um projeto pode estar ligado às tecnologias que o cercam. Os sistemas operacionais e as linguagens de programação podem influir diretamente no projeto. Se o conhecimento dessas tecnologias influir diretamente no sucesso do projeto elas deverão ser classificadas como informações. (US ARMY, 2003).

- Critérios de aceitação do produto: O cliente pode exigir critérios para aceitar o projeto. Se existir a dúvida para determinar quais são os critérios de aceite do cliente, classifique os critérios de satisfação como informações.

- Necessidades externas: As necessidades externas de testes operacionais ou decisões executivas devem ser classificadas como informações.

- Experiência: A experiência da equipe do projeto pode identificar possíveis dificuldades com base em projetos anteriores. Estas identificações devem ser classificadas como informações.

- **Mapear Informações**

Com as informações colhidas no passo anterior, a próxima tarefa é mapear as informações de acordo com as áreas de informação do PSM. O objetivo do mapeamento é criar sete grupos comuns de informações. São eles:

- Schedule e Progresso – Informações sobre prazos.

[18] Processo que avalia os riscos de um projeto. São utilizadas diversas técnicas para esta avaliação. Para maiores informações acesse : http://www.simpros.com.br/Apresentacoes_PDF/Artigos/Art_24_Simpros2004.pdf

- Recursos de Custo – Informações sobre orçamentos e custo.

- Tamanho do produto e Estabilidade – Informações de estabilidade e tamanho do produto.

- Qualidade do produto – Informações de necessidades do cliente. A qualidade do projeto esta ligada diretamente ao cumprimento das necessidades informadas pelo cliente.

- Desempenho do processo – Informações sobre o desempenho do projeto. O desempenho do projeto está ligado às entregas de produtos de fornecedores e terceiros para o desenvolvimento do projeto.

- Eficácia da Tecnologia – Informações sobre a viabilidade de tecnologias novas, e de tecnologias necessárias para o desenvolvimento do projeto.

- Satisfação do cliente – Informações sobre a satisfação do cliente. Geralmente as informações de satisfação do cliente são as perspectivas de resultados do projeto.

- **Priorizar Informações**

Com as informações agora divididas em grupos comuns, o próximo passo é determinar quais serão as prioridades das informações. Para determinar a prioridade são atribuídos a cada informação dois valores, um de probabilidade de ocorrência e um de impacto. Estes valores são multiplicados obtendo-se, assim, um valor de exposição do projeto. Quanto maior for o índice de exposição maior será a prioridade daquela informação. (US ARMY, 2003).

3.2.2.2. Selecionar e especificar métricas de projeto

Com base nas informações colhidas na atividade identificar e priorizar necessidades da informação; as métricas serão especificadas. A figura 18 demonstra os processos para a identificação das métricas.

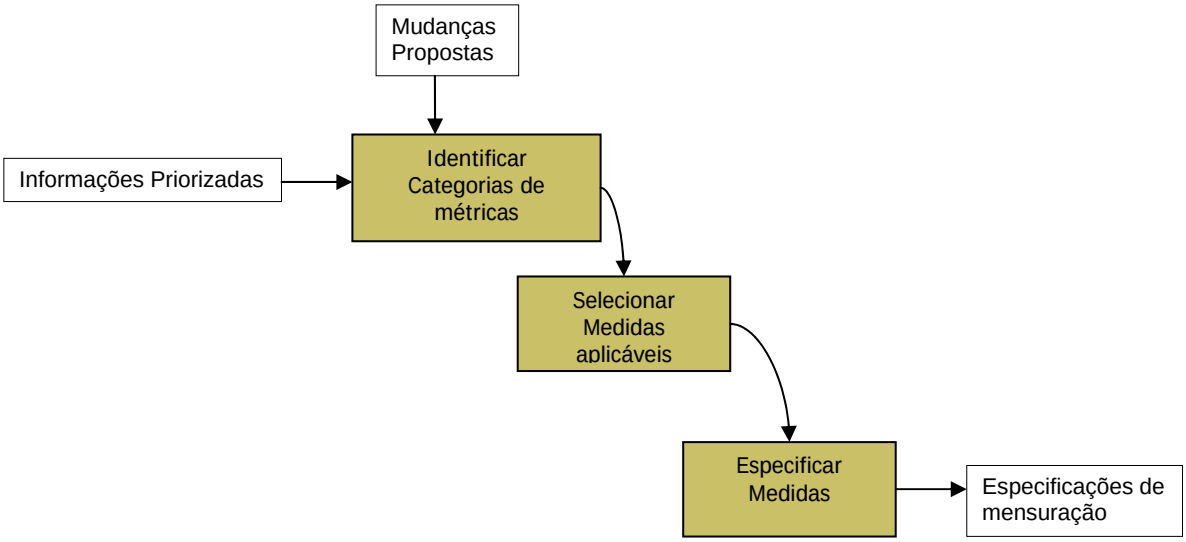


Figura 18 - Selecionar e especificar métricas de projeto
Fonte: Traduzido do PSM - US. ARMY (2003)

- **Identificar categorias de métricas**

Após determinadas as áreas comuns de informações, são identificadas as categorias de métricas que deverão ser aplicadas em cada área. Esta identificação é dada por uma tabela[19] que classifica as categorias de métricas de acordo com as áreas de informação.

- **Selecionar medidas aplicáveis**

Determinadas as categorias de métricas se faz necessário definir quais as melhores métricas que se aplicam ao projeto. Existem muitas métricas, por este motivo elas podem ser aplicadas em áreas de informação diferentes, o recomendável é que se utilize o número máximo de métricas, pois quanto mais dados tivermos, mais preciso o projeto ficará.

A definição de métricas não é uma tarefa prática, por este motivo, geralmente, selecionam-se as métricas que têm uma maior aproximação com a área de informação, identificando o melhor conjunto de medidas que atendem ao projeto. Elas são selecionadas baseadas em:

- Medida de eficácia.
- Domínio de características.
- Práticas de administração de projeto.

[19] Tabela de classificação de métricas em Anexo B .

- Custo e disponibilidade.

- Ciclo de vida.

- Necessidades externas.

- Tamanho/origem de sistemas externos.

O PSM fornece tabelas que auxiliam a seleção das medidas; essas tabelas podem ser consultadas no *Practical Software and Systems Measurement – 4.0c,* Capitulo 3 - *Measurement Selection and Specification Tables*[20].(US ARMY, 2003).

- **Especificar medidas**

Selecionadas as medidas aplicáveis é necessário especificá-las, pois uma única medida pode possuir interpretações diferentes, que são determinadas pelas empresas e seus engenheiros. É importante especificar o que realmente cada medida faz, para que não ocorra dupla interpretação sobre seu uso.

As especificações de uma medida devem possuir:

- Itens de dados: Definir o que será contado na medida. Exemplo: horas de trabalho.

- Atributos: São propriedades da medida ou do que esta sendo medido. Geralmente os atributos são utilizados para relacionar os itens de dados.

- Estrutura de agregação: É a maneira que o projeto esta organizado para sua medida total. Existem três estruturas de agregação.

  - Agregação baseada em componentes: O projeto está dividido de acordo com os componentes existentes.

  - Agregação funcional: Os componentes do projeto estão divididos nas suas áreas especificas.

  - Agregação ativa: Os componentes do projeto estão divididos nas áreas do desenvolvimento do projeto.

[20] *Practical Software and Systems Measurement – 4.0b* – Disponível no Anexo XX em CD

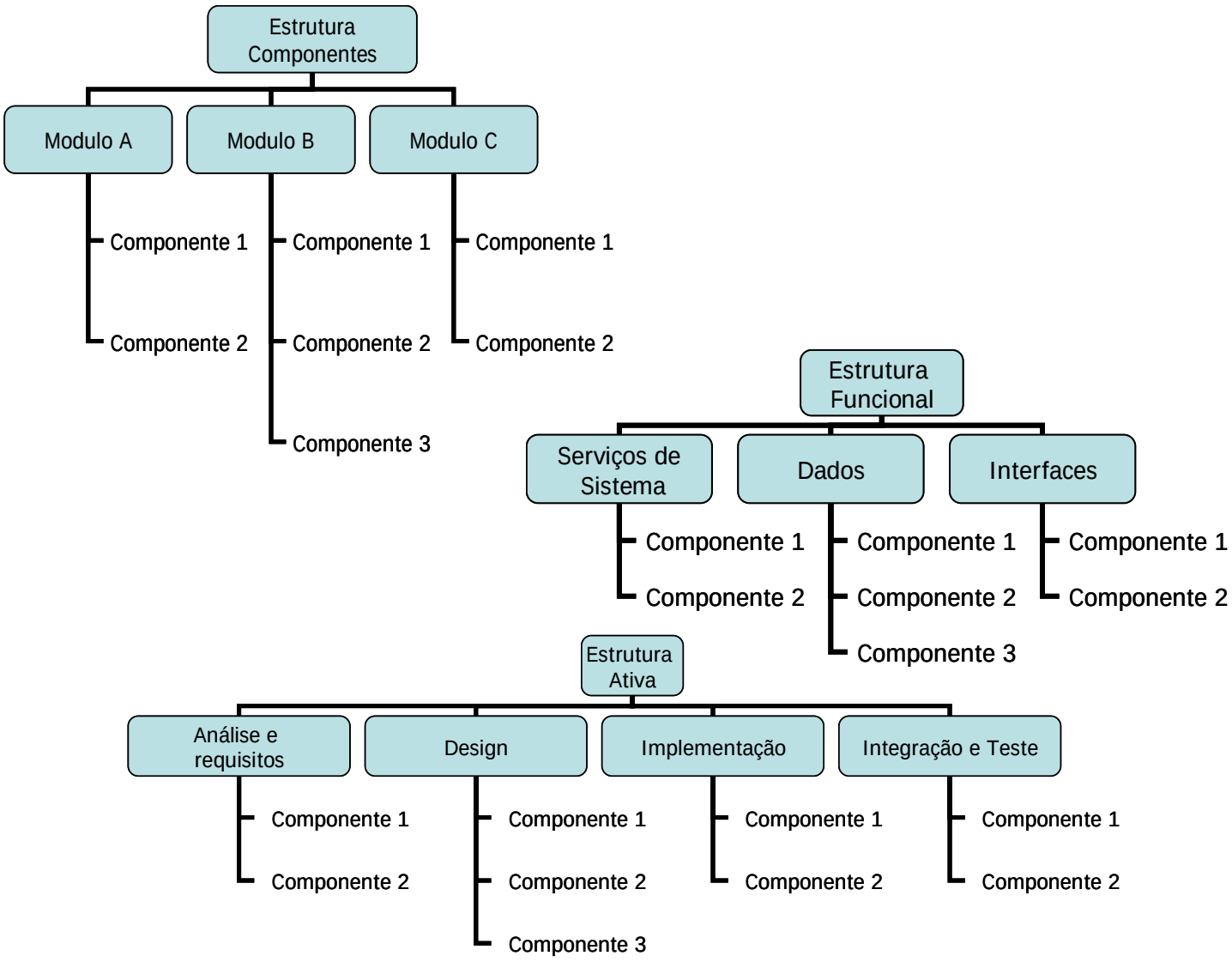


Figura 19 - Estruturas de agregação
Fonte: Traduzido do PSM - US. ARMY (2003)

3.2.2.3. Integrar mensuração aos processos do projeto

Até este ponto, o processo de planejar a mensuração foi focado em classificar as informações e medidas do projeto. A função desse item é examinar como o processo de medida esta sendo útil para o projeto. Nem sempre uma grande massa de dados pode retratar as reais necessidades de um projeto.

Integrar mensuração aos processos do projeto, esta dividida em:

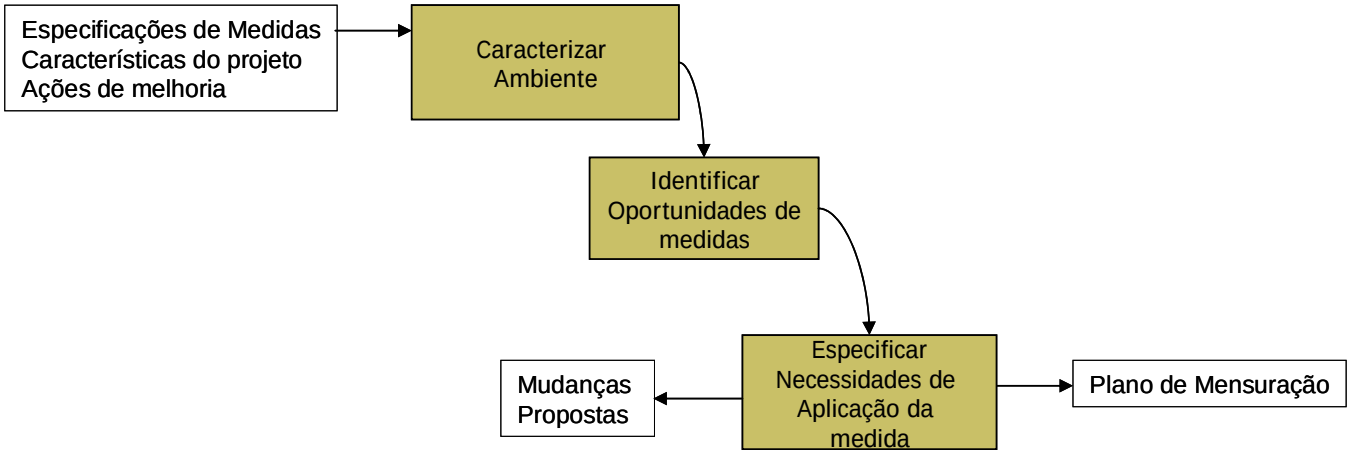


Figura 20 - Integrar mensuração aos processos do projeto
Fonte: Traduzido do PSM - US. ARMY (2003)

- **Caracterizar Ambiente**

A definição do plano de mensuração não pode basear-se somente nas necessidades técnicas de informação. É importante que sejam considerados os processos de administração e ambiente do projeto.

Ao caracterizar ambiente busca-se avaliar as condições e técnicas em que o projeto está sendo construído (ambiente de construção).

Alguns fatores-chave para a caracterização do ambiente são:

- Qual modelo de ciclo de vida ou estrutura utilizado?

- Estrutura de término do produto. Qual será a estrutura de término do produto? Quais incrementos foram definidos e quantos profissionais foram alocados para as tarefas?

- Quais atividades estão sendo medidas?

- Quais tecnologias estão sendo utilizadas? Linguagens de programação? *Design?* Ferramentas?

- Quais os planos de softwares e hardwares?

- Como está sendo aplicada a administração do projeto?

- Qual processo de engenharia está sendo aplicado?

- Qual o nível de maturidade do processo?

- Quais são os prazos?

Para controlar estas informações e determinar o ambiente é possível utilizar o diagrama *Work brakedown struture* (WBS)[21]. Nele se pode classificar as atividades e a estrutura em que estão ocorrendo às tarefas no projeto. A coleta das informações e a caracterização do ambiente se tornam mais fáceis de determinar.

A caracterização do ambiente é muito importante, pois o ambiente influi diretamente nas fases do projeto. Um ambiente mal estruturado pode tornar as medidas do projeto não confiáveis, já que o ambiente pode alterar o processo; exemplo: mudança de uma linguagem de programação. (US ARMY, 2003).

- **Identificar Oportunidades de Medida**

Durante a fase de planejar a mensuração, é importante explorar mecanismos de métricas já aplicados em outras empresas ou em outros projetos. O uso de mecanismos já existentes traz a vantagem de baratear o processo de medidas, além de otimizá-lo. A fase de identificar oportunidades de medida tem o objetivo de buscar soluções já desenvolvidas para auxiliar o novo planejamento de medidas.

Os dados de medidas são originados por muitas fontes e que se classificam em três tipos:

- Dados Históricos – São dados colecionados de projetos passados. Este tipo de dado ajuda na projeção de orçamentos e estudo de viabilidade de planos.

- Dados de planejamento – São dados de orçamentos e esforço do projeto. Controlam alterações e o progresso do projeto.

- Dados de desempenho atual – São dados do próprio projeto. Através da evolução do projeto é possível utilizar dados anteriores para comparar com os dados atuais.

- **Especificar necessidades de aplicação das medidas**

Nesta fase é efetuada análise de todos os dados coletados, desde a fase de identificação de informações até a identificação de oportunidades de medida. Todos estes procedimentos precisam ser definidos antes que seja construído o plano de mensuração.

Esta fase é uma combinação de definições operacionais e procedimentos que guiam a construção do plano. Para isto as seguintes atividades precisam ser focadas:

[21] Fluxo de tarefas.

- Definições de medida – Revise todas as fases de especificação de medidas. Descreva os métodos das medidas, incluindo seus critérios de avaliação (conforme descrito na fase: especificar medidas). Documente todo o processo.

- Escopo da medida – Classifique os escopos de cada medida. Existem medidas que podem ser amplamente utilizadas, porém, é necessário determinar suas limitações, dentro do projeto. Documente a duração e as fases do ciclo de vida da medida dentro do projeto.

- Coleta de dados – Documente o processo de coleta de dados. Crie um portfolio de medidas para que ele possa ser utilizado neste projeto e em projetos futuros. Defina uma estrutura para coleta de dados; exemplo: coletas semanais.

- Análise de dados – Defina indicadores básicos para serem gerados periodicamente e, através de seus resultados, poder determinar como o projeto esta caminhando.

- Informação de resultados – Defina relatórios que apresentem a extração de dados do projeto. É recomendável a criação de dois relatórios: um técnico para os profissionais envolvidos e um administrativo para os executivos do projeto.

- Avaliação de medida – O planejamento de medida deve ser reavaliado a cada trimestre ou semestre. Esta avaliação garante a adaptação do planejamento para que as métricas continuem retratando a real situação do projeto.

Após reavaliados todos os processos é construído o plano de mensuração. Sua estrutura é apresentada no Quadro 15, que segue.

PLANO DE MENSURAÇÃO
Parte 1 – Introdução
Propósito do projeto (Objetivos)
Parte 2 – Descrição do projeto
Características técnicas e administrativas do projeto
Parte 3 – Regras de mensuração, responsabilidades, comunicação
Como a medida será empregada no processo técnico e administrativo Pontos de contato de medição (Contratações, recursos) Responsabilidades das métricas Comunicação da empresa Ferramentas de bancos de métricas
Critérios de avaliação
Parte 4 – Descrição das informações do projeto
Informações da empresa e do projeto Lista de informações prioritárias
Parte 5 – Especificação das Métricas
Para cada medida selecionada para o projeto, defina: - Nome - Categoria de informação e de métrica - Itens de dados - Atributos - Estrutura de agregação - Nível - Critérios - Definição de dados - Estimativa baseada em metodologia, e dados históricos - Mecanismos de relatório - Freqüência de aplicação - Fases do projeto que será aplicada
Parte 6 – Estrutura de agregação do projeto
Estrutura por componentes Estrutura funcional Estrutura funcional
Parte 7 – Indicadores
Para cada indicador selecionado, defina: - Nome - Área de informação correspondente - Medidas utilizadas nessa área - Decisões que poderão ser tomadas diante aos resultados deste indicador.
Parte 8 – Relatórios
Freqüência da emissão de relatórios; Estrutura dos relatórios;

Quadro 15 - Plano de Mensuração
Fonte: Traduzido do PSM - US. ARMY (2003)

3.2.3. Executar mensuração

A fase de executar a mensuração tem o objetivo de executar as métricas planejadas. Os resultados das métricas devem responder às questões de informação do projeto.

A figura 21 demonstra o fluxo das tarefas necessárias para executar a mensuração.

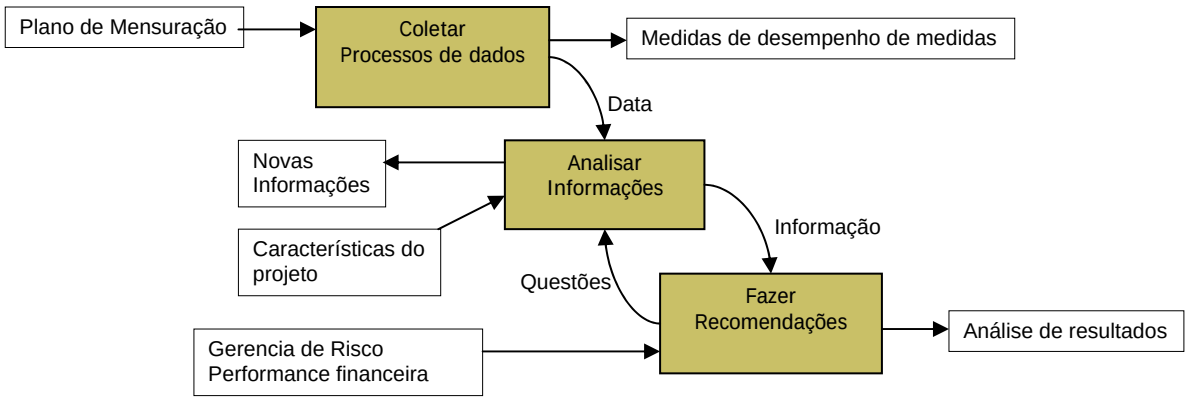


Figura 21 – Execução de mensuração
Fonte: Traduzido do PSM - US. ARMY (2003)

3.2.3.1. Coletar processos de dados

Coletar e entender os dados é a primeira tarefa da execução de métricas. Eles devem refletir a natureza do produto, e a real situação do projeto. O processo de coleta de dados e suas respectivas fases são apresentados na figura 22:

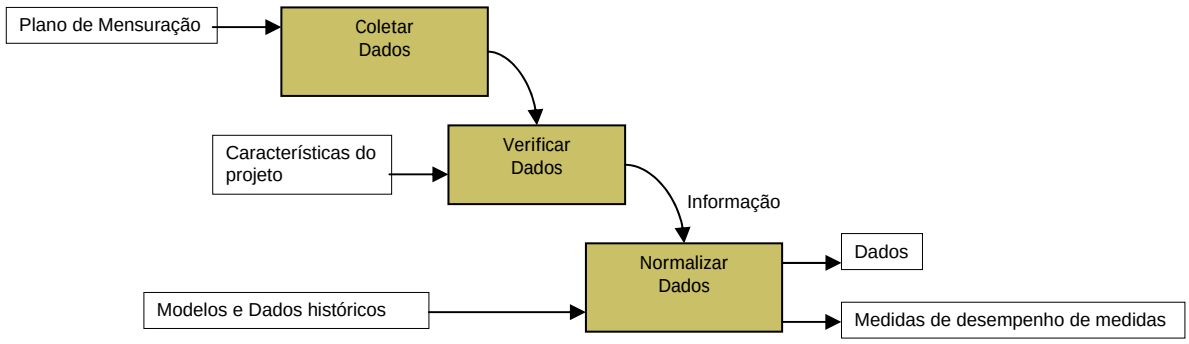


Figura 22 – Coleta de processos de dados
Fonte: Traduzido do PSM - US. ARMY (2003)

- **Coletar dados**

O processo de coleta de dados é executado pelo analista de medidas.

Ele colhe os dados seguindo as especificações do plano de mensuração (desde as técnicas de coleta de dados até os formatos dos dados).

A coleta de dados é feita através dos mecanismos de acesso aos dados, que são:

- Direto; acesso compartilhado – O analista de medida tem acesso às fontes de dados de projeto, diretamente.

- Cópias eletrônicas de banco de dados ou de arquivos – A equipe de projeto fornece ao analista de métricas, cópias dos bancos de dados e dos arquivos do projeto. O analista de medida coleta os dados através das cópias.

- Exportação de dados – A equipe de projeto gera relatórios digitais com informações especificas para o analista de medida, geralmente relatórios seguindo um padrão já estabelecido. O analista de medida trata os dados em uma ferramenta de análise.

- *Hardcopy* (Cópia) – A equipe de projeto gera relatórios impressos. O analista de medida tem que fazer a coleta e a análise dos dados manualmente.

- **Verificar dados**

A previsão das métricas é totalmente dependente da qualidade dos dados coletados.

Os dados devem ter uma fonte segura e devem ser exatos para que a métrica consiga analisar corretamente o processo do projeto.

A verificação dos dados é um processo complicado, por existirem diversas maneiras de interpretá-lo; o projeto também pode assumir diferentes estruturas de desenvolvimento. É preciso garantir que os dados sigam sempre as mesmas especificações, para que sejam sempre confiáveis para a aplicação das métricas.

Uma maneira de verificar os dados é criar um *ckeck-list*[22], que verifique se os dados atendem as exigências para se tornar confiável. (US ARMY, 2003)

[22] Verificações de itens.

- **Normalizar dados**

Agora que as fases de coleta e execução das métricas estão concluídas é necessário normalizar os dados. A normalização é o processo de converter os dados de diferentes métricas em uma unidade de medida comum.

A padronização de uma única unidade de medida para o processo de métricas facilita a analise dos resultados obtidos.

3.2.3.2. Analisar Informações

Os resultados das métricas se tornam novas informações para o projeto. Os administradores do projeto devem analisar estas informações para determinar quais ações devem ser tomadas para o sucesso do mesmo.

A tarefa de analisar informações aplicará as métricas seguindo o modelo apresentado na figura 23.

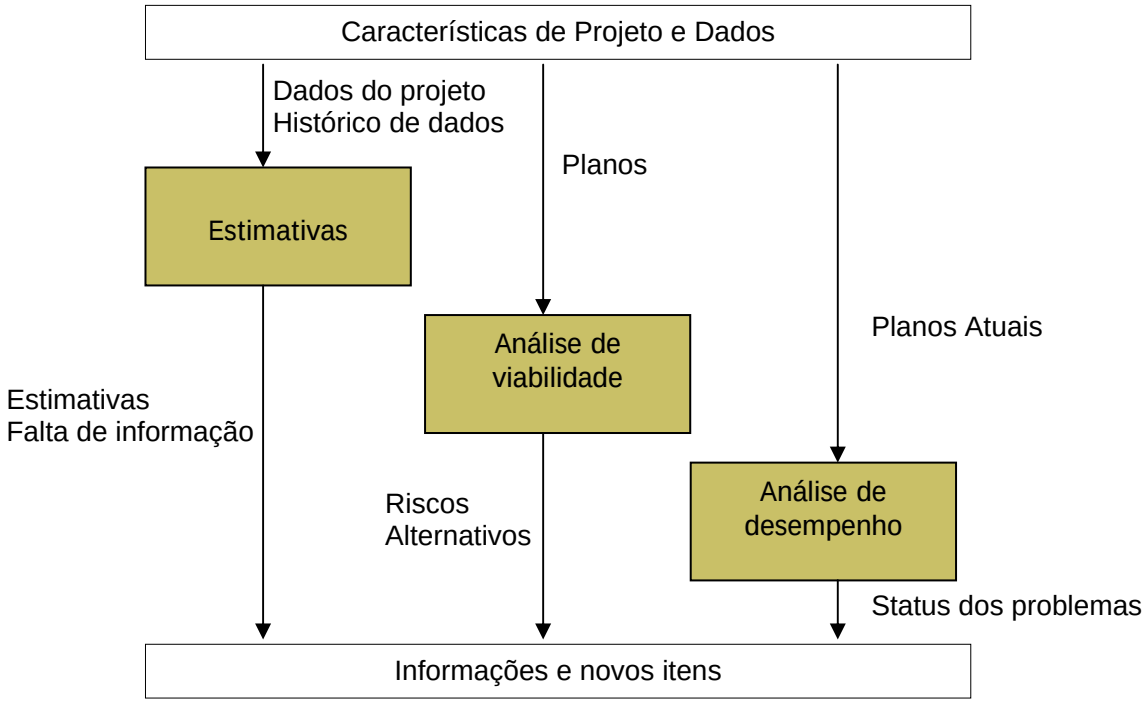


Figura 23 – Análise de informações
Fonte: Traduzido do PSM - US. ARMY (2003)

- **Estimativas**

Estimativas são um tipo de análise numérica que baseiam-se nos dados atuais para produzir projeções dos dados futuros. Esta análise é tipicamente utilizada para estimar valores de métricas de tamanho, esforço e orçamento.

É necessário tomar muito cuidado com a utilização das estimativas, pois os resultados podem ser imprecisos e, assim, prejudicar orçamentos e prazos pré-definidos. O principal fator que torna a estimativa fraca é a falta de informação; o cálculo necessita de dados passados para projetar dados futuros, se a base de informações for fraca não há como estimar um valor com precisão.

O PSM possui uma tarefa de cálculo de estimativas, que é apresentada na figura 24.

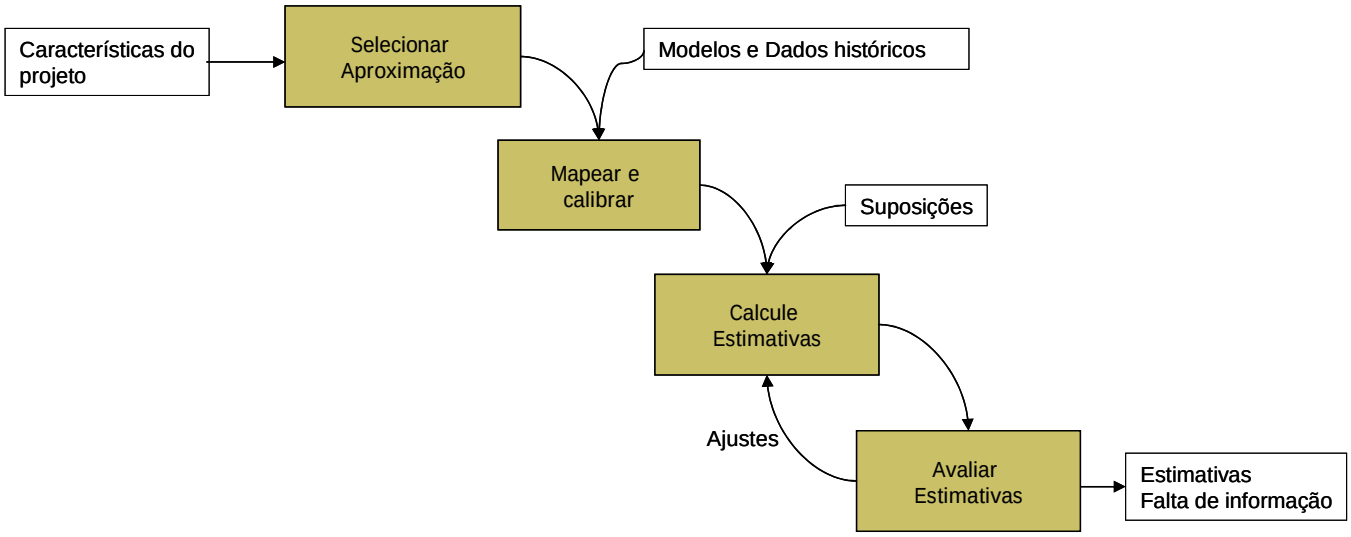


Figura 24 - Tarefas do processo de estimativas
Fonte: Traduzido do PSM - US. ARMY (2003)

- *Selecionar Aproximação (alinhamento)*

A tarefa de selecionar aproximação é responsável pela seleção da métrica que será utilizada.

As métricas estatísticas são modelos matemáticos que estão divididos em quatro grupos mostrados no Quadro 16.

Estimativa	Descrição	Dados necessários	Complexidade Matemática
Modelos Paramétricos	Descreve a informação Geral	Vários projetos que Utilizaram o mesmo modelo	Técnicas de estatística complexa.
Modelo baseado em atividades	Detalha a informação do processo e do produto	Dados detalhados de vários projetos	Aritmética
Analogia	Detalha a informação do produto	Pelo menos um projeto semelhante	Aritmética
Relações de estimativa simples	Descreve a informação Geral	Vários projetos	Técnicas de estatística simples.

Quadro 16 - Modelos Matemáticos
Fonte: Traduzido do PSM - US. ARMY (2003)

- Mapear e calibrar

Terminada a tarefa de aproximação é necessário mapear e calibrar os dados estatísticos para o atual projeto. Esta tarefa é muito importante, pois cada projeto possui suas próprias características; as métricas precisam estar calibradas para que o resultado não seja distorcido.

A utilização de estimativas não garante orçamentos precisos; é preciso ter muita cautela ao selecionar os dados e calibrá-los, pois vários são os fatores que podem alterar um resultado. Os pontos que devem ser considerados ao calibrar os dados são:

- Fonte de Dados: Quais são as características da fonte de dados? Verifique quais problemas o projeto origem dos dados sofreu.
- Âmbito de aplicação: Qual é o ciclo de vida do projeto origem dos dados? É recomendável que os dados sejam trabalhados por fase do ciclo de vida e não do projeto total.
- Nível de detalhe: Qual é o nível de detalhe dos dados? Procure entender a estrutura de agregação dos dados.
- Esforços de atributo: Como foi efetuado o cálculo de esforço?
- Programação de atributos: Em quais condições as programações foram efetuadas? Verifique a fase e as condições do projeto para avaliar os tempos do cronograma.
- Tamanho dos atributos: Verifique como foram gerados os atributos. É importante avaliar a maneira que o atributo foi construído, pois isto influi diretamente na complexidade do software. A linguagem de programação deve ser considerada.
- Déficit de atributos: Qual é a qualidade do atributo? Verifique quais problemas o atributo sofreu e quanto de sua qualidade foi sacrificada em troca de esforço.

- Calcular estimativas

Com o modelo matemático definido e os dados calibrados para o projeto, agora são executadas as métricas. Este processo é definido em quatro classificações de características do projeto a serem atendidas pelas métricas. São elas: tamanho, esforço, prazo e qualidade. (US ARMY, 2003).

- Tamanho

As estimativas de tamanho calculam o tamanho de um software. As duas técnicas mais utilizadas são: a contagem de linhas de código e pontos de função (ambas medidas já foram apresentadas no capitulo Medidas de software). O resultado destas métricas informará, antes de o projeto ser concebido, qual será seu tamanho estimado. Este valor é requisito para o calculo de outras métricas como esforço, orçamento e prazos.

- Esforço

As estimativas de esforço calculam o esforço e a mão de obra necessária para a construção do projeto. O cálculo do esforço é dado por uma equação geral, pela Lei de Zipf ou pelos modelos de PERT (mostrados no capítulo 2).

- Prazo (Cronograma)

As estimativas de prazo são calculadas pelo conjunto dos resultados de esforço e tamanho do software. Os modelos paramétricos como COCOMO, SEER (mostrados no capítulo 2) são utilizados para a estimativa de prazos.

- Qualidade

As estimativas de qualidade são objetivos de qualidade pré-definidos para o projeto. As métricas são utilizadas para avaliar se os objetivos de qualidade estão sendo cumpridos. Geralmente, o calculo de qualidade esta relacionado com a contabilização de problemas que o software sofreu e pelo seu desempenho.

<u>- Avaliar estimativas</u>

Avaliar estimativas ocorre em três perspectivas:

- Satisfação de restrições: É o planejamento das métricas de prever as restrições do projeto. Caso durante o desenvolvimento ocorram mudanças de restrições (orçamentos, prazos), as estimativas deverão ser reavaliadas e as alterações deverão ser documentadas.

- Documentação das estimativas: A boa documentação aumenta a qualidade e facilita a estimativa de dados. Durante o processo será necessário a re-estimativa de algumas variáveis; com uma boa documentação é possível determinar com precisão as informações que sofreram alteração.

- Qualidade das estimativas: O processo de estimativas é reavaliado para determinar se todas as tarefas foram cumpridas para garantir a qualidade. São reavaliados itens como: origem dos dados, atributos, calibração dos dados, etc.

- Análise de viabilidade

A tarefa da análise de viabilidade é avaliar o realismo dos resultados das métricas. Seu objetivo é garantir que as estimativas sejam realistas, validar o plano de estimativas ou produzir novas alternativas.

A viabilidade de um projeto depende da exatidão dos dados utilizados para o cálculo das estimativas e do planejamento de desenvolvimento do projeto. Por isto o primeiro item a ser analisado pela viabilidade é a condição do plano de projeto. Devem ser considerados os itens:

- O plano está completo?
- Os recursos foram previstos corretamente? (duração do projeto, feriados, férias, afastamento de profissionais).
- Os recursos estão disponíveis? (softwares, hardwares, documentação, módulos).

Após a avaliação do plano de projeto, os dados utilizados pelas métricas e os resultados das métricas deverão ser avaliados, considerando os itens:

- Confiabilidade dos dados de origem: Novamente é necessário verificar a confiabilidade dos dados que foram utilizados pelas métricas.
- Estabelecer uma comparação entre os resultados das métricas do projeto atual com projetos antigos. Os resultados devem ser razoavelmente semelhantes, quando excluídos defeitos ou problemas dos projetos.

As análises de viabilidade devem ser aplicadas sempre que o projeto sofrer mudanças significativas como:

- Alterações de funcionalidades.
- Alterações organizacionais.

- Alterações tecnológicas. (US ARMY, 2003).

A tarefa de análise de viabilidade procura tornar as estimativas, o mais real possível; durante esta avaliação poderão ser identificados problemas e será necessário propor uma alternativa para sua solução. Esta alternativa pode ser o re-cálculo da estimativa ou a adequação das tarefas. (US ARMY, 2003).

- Análise de desempenho

A tarefa de análise de desempenho avalia se os processos do projeto estão ocorrendo da maneira e no tempo que foram projetados. O objetivo da análise é de prover informações para que o gerente de projeto possa tomar decisões antes que o projeto comece a divergir do plano. Sua prática no projeto deve ser periódica.

O processo de análise de desempenho é composto de 4 tarefas:

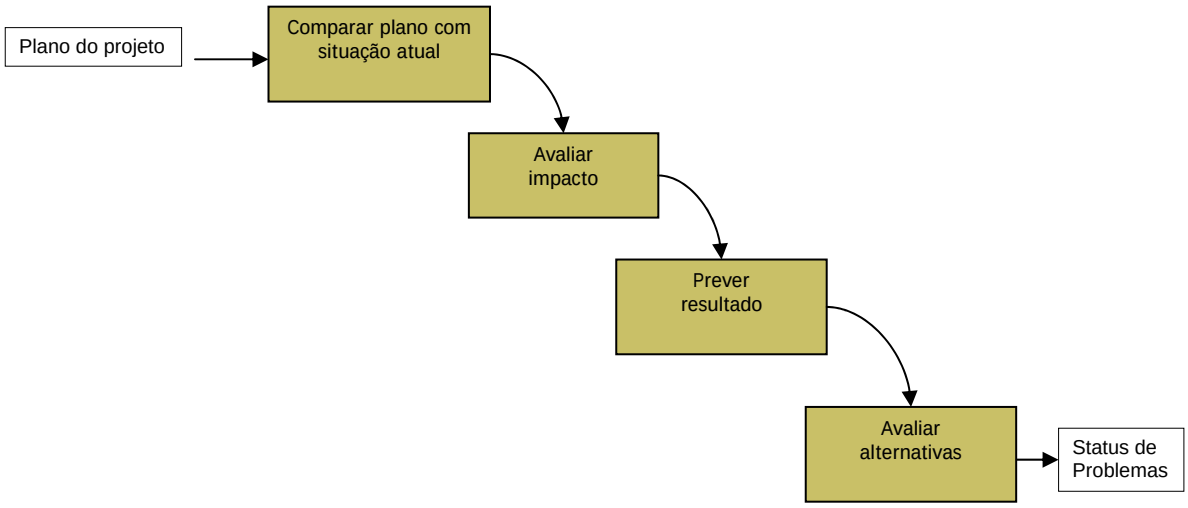


Figura 25 - Análise de desempenho
Fonte: Traduzido do PSM - US. ARMY (2003)

- Comparar plano com situação atual

Os resultados obtidos na análise de desempenho devem ser comparados com os valores predefinidos no plano do projeto. Eventuais distorções de valores poderão ocorrer isoladamente, não prejudicando o cumprimento do projeto. Quando uma variável como, por exemplo os prazos de desenvolvimento, começa a desviar do resultado previsto em várias medições de performance, esta situação é classificada como um problema para o projeto,

sendo necessário definir ações para que esta variável retorne aos valores previstos. (US ARMY, 2003).

- Avaliar impacto

Com a identificação de problemas no processo do projeto é imperativo avaliar o impacto deles ao objetivo e cumprimento do projeto.

Para identificar o impacto do problema ao projeto deve-se:

- Localizar a fonte de dados que estão gerando o problema.
- Verificar se ocorreu alteração de âmbito de aplicação (plano X atual).
- Colher dados adicionais quando necessário.
- Avaliar quais tarefas estão ligadas a este problema.
- Determinar se o prazo e o custo do projeto sofreram impactos.

Após esta análise é possível determinar qual é o real impacto do problema ao projeto e, assim, tomar ações para que o problema seja solucionado.

- Prever resultado

A tarefa de prever resultado tem a função de estimar o resultado final do projeto, caso ele continue com este problema. Esta projeção é importante, pois ela mostra quais serão as conseqüências de um problema no resultado do projeto. Existem problemas que sua previsão de impacto é mínima, não sendo necessária sua reestruturação, porém, existem problemas que se não forem reestruturados no início, poderão levar o projeto ao fracasso. (US ARMY, 2003).

- Avaliar alternativas

Caso o problema seja prejudicial ao cumprimento do objetivo do projeto será preciso avaliar alternativas para a solução do problema. Para esta análise de ser considerado:

- Os dados históricos e experiências em projetos antigos.
- A avaliação de reestruturação de prazos e custos.
- A identificação de novas informações (Planejar a mensuração).

Se for possível, a remarcação dos prazos é a medida menos complexa, porém, se isto não for possível, às vezes, poderá ser necessário reavaliar o planejamento e efetuar alterações como: inclusão e exclusão de funcionalidades para que o problema possa ser solucionado. (US ARMY, 2003).

3.2.3.3. Fazer recomendações

O propósito do PSM é auxiliar gestores de projeto a tomarem melhores decisões. A tarefa final do processo de medida central é informar os resultados obtidos e fazer recomendações.

O andamento do processo de medidas e os resultados são informados através de relatórios periódicos, que devem conter:

- Avaliação global do projeto: Informa as estimativas e o desempenho do projeto atual. Demonstra como está o desempenho do projeto.

- Identificação de problemas: Informa os problemas identificados e suas causas : riscos, falta de informação, impacto.

- Recomendações: Informa alternativas para a solução de problemas. Avalia as vantagens e desvantagens de novas tecnologias que poderão ser aplicadas ao projeto.

- Possibilidades de novos problemas: Informa quais problemas poderão ser gerados no futuro do processo.

É recomendável que os relatórios sejam avaliados pela equipe de projeto antes de serem encaminhados para a diretoria executiva da empresa. A equipe de projeto faz a última avaliação e confirma se as informações que estão registradas no relatório retratam a realidade do projeto. (US ARMY, 2003).

3.2.4. Avaliar Mensuração

É improvável que na primeira aplicação de um programa de medidas, este seja realizado perfeitamente. A experiência em programas anteriores e a avaliação da mensuração são fatores que tornaram a aplicação de um plano de medidas mais precisa.

O processo de avaliação de medidas é composto de quatro tarefas, demonstradas na figura 26.

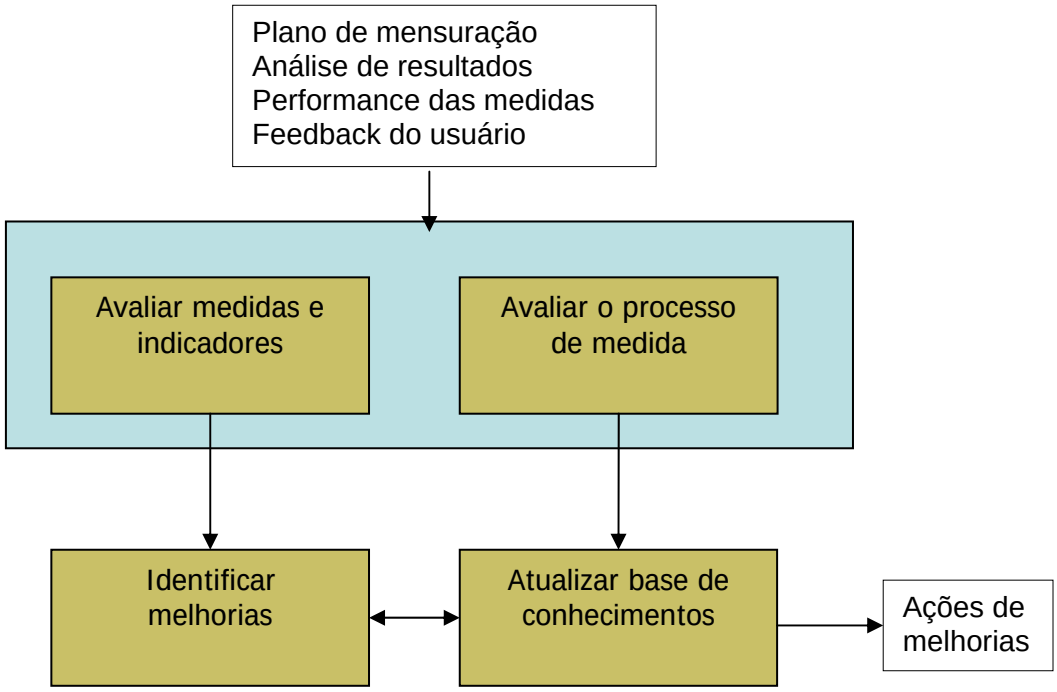


Figura 26 – Avaliação de mensuração
Fonte: Traduzido do PSM - US. ARMY (2003)

- Avaliar medidas e indicadores

Para a avaliação das medidas e indicadores existem critérios pré-definidos pela ISO/IEC – 15939 – Processo de medida de software, juntamente com o CMMI.

Estes critérios são:

- Resultados da utilização de medida: Este critério mede até que ponto o gerente do projeto ou o analista de medida, utiliza as métricas. Como as métricas estão sendo empregadas? Todas são úteis para o projeto?

- Confiança nos resultados de medida: Este critério avalia a confiança que os usuários têm sobre os resultados das medidas. A participação dos usuários nos processos de medidas e a utilização de feedback são importantes para aumentar a confiança nos resultados.

- Aptidão da medida diante ao projeto: Este critério avalia se a medida realmente está satisfazendo as necessidades de informação. Nem todas as medidas trabalham bem em todas as situações. A medida esta sendo empregada corretamente? Esta sendo aproveitada pelo projeto?

- Resultados de medida: Este critério avalia a facilidade com que o analista de medida entende os indicadores do projeto.

- Satisfação das suposições do modelo indicador: Este critério avalia qual o grau de satisfação quando da utilização dos dados históricos de outros projetos. Os dados históricos utilizados atenderam as necessidades do projeto?

- Exatidão da medida: Este critério avalia como a medida foi aplicada, se seus requisitos e procedimentos foram executados corretamente.

- Confiança na medida: Este critério avalia a consistência dos resultados de uma medida. Esta confiança é avaliada por dois fatores:

  - Repetição: Quantas vezes esta medida foi utilizada pela empresa, nas mesmas condições? Quanto maior o número de utilizações, mais confiável a medida será.
  - Reprodução: Quantas vezes esta medida foi utilizada pela empresa, em condições diferentes e adaptadas?

- Avaliar o processo de medida

A avaliação do processo de medida é dividida em três características: desempenho, auditoria e capacidade. É recomendável que a avaliação de desempenho e conformidade ocorra a cada seis meses e a de capacidade a cada dois anos. (US ARMY, 2003).

- Desempenho

O próprio processo de medida deve ser medido. É preciso avaliar o desempenho que a medida tem sobre o projeto e quais são seus benefícios. Para fazer esta avaliação deve-se obedecer a quatro critérios:

- Oportunidades
- Eficiência
- Enquadramento
- Satisfação do cliente

- Auditoria

É necessário que uma auditoria externa avalie o processo de medida. A auditoria externa tem a capacidade de avaliar e identificar problemas não reconhecidos pela equipe de medidas. A equipe de auditoria se baseia nas especificações do ISO/IEC – 15939.

- Capacidade

Os modelos ISO/IEC 15939 e o PSM descrevem as tarefas básicas que devem ser executadas em qualquer processo de medida. Porém, com a utilização do processo de medida,

novas tarefas mais detalhadas surgem elevando, assim, o nível de capacidade da empresa para a aplicação de métricas de software. (US ARMY, 2003).

- *Identificar melhorias*

As melhorias identificadas em cada projeto devem ser armazenadas (documentadas) para projetos futuros. Em todas as fases do PSM a equipe de medidas encontrará possíveis melhoras. Esta tarefa tem o objetivo de catalogar as melhorias através do escopo do PSM.

Melhorias de:

- Planejar a mensuração
- Executar a mensuração
- Avaliar a mensuração

- *Atualizar base de conhecimento*

Todas as informações obtidas nas fases: planejar mensuração, executar mensuração, e analisar mensuração deverão alimentar a base de conhecimento da empresa para projetos futuros. (US ARMY, 2003).

A base de conhecimento deve conter:

- Planos de medidas, políticas e procedimentos.
- Definição de medidas e indicadores.
- Dados de verificações técnicas.
- Satisfação do cliente.
- Relatórios executados.
- Auditorias executadas.
- Nível de capacidade.
- Problemas e soluções.

3.3. PSM INSIGHT

O PSM INSIGHT é uma ferramenta desenvolvida pelo departamento de defesa norte americano, juntamente com o ASMO (*Army Software Metrics Office),* para a implantação do modelo PSM. Atualmente na sua versão 4.2.2, a ferramenta auxilia profissionais para a aplicação das técnicas PSM e para o acompanhamento de métricas. (US ARMY, 2003)

3.3.1. Visão geral da ferramenta

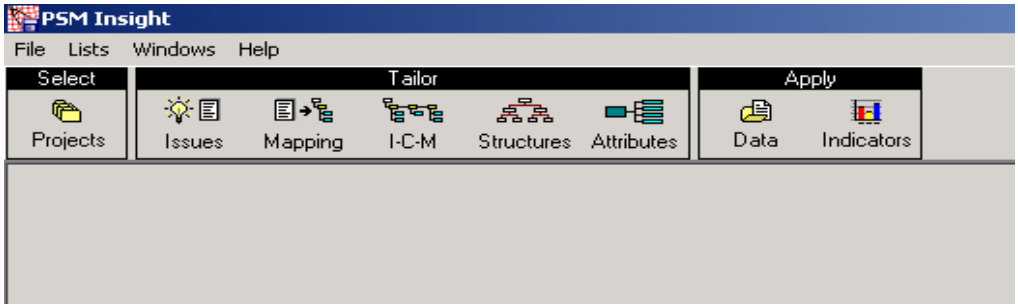


Figura 27 - PSM INSIGTH
Fonte: Traduzido do PSM - US. ARMY (2003)

- *Select Project*: Menu utilizado para criar ou abrir novos projetos no PSM INSIGHT.

- *Tailor*: Atividades de Planejar a mensuração.

- *Issues*: Menu responsável pelo cadastro de informações críticas do projeto.

- *Mapping*: Menu responsável pela classificação das informações criticas, nas sete áreas do PSM.

- *I-C-M: Issues – Categories – Measures*: Menu responsável pela priorização das informações críticas e pela seleção das categorias e métricas que serão utilizadas.

- *Strutures*: Menu responsável pelo cadastro das estruturas de agregação do projeto.

- *Attributes*: Menu responsável pela classificação dos atributos.

- *Data*: Menu responsável pela alimentação de dados para o sistema.

- *Indicators*: Menu responsável pela criação de gráficos e indicadores das métricas aplicadas.

- *Reports*: Todos os menus têm a função de emissão de relatórios. Para acessar esta função clique com o botão direto do mouse sobre o item que deseja extrair um relatório. (US ARMY, 2003)

Considerando o apresentado neste capítulo sobre o PSM INSIGHT, partimos para a sua implementação na empresa INFRAERO.

CAPÍTULO IV - ESTUDO DE CASO: INFRAERO

Neste capítulo é apresentado um estudo de caso desenvolvido na Empresa Brasileira de Infra-estrutura Aeroportuária – INFRAERO[23], uma empresa pública vinculada ao Ministério da Defesa.

O estudo de caso fornece um formato para documentação de descobertas da avaliação Pós-projeto. O relatório deve ser completado pelo Gerente de Projetos, recorrendo se necessário a um Consultor do PMO quando houver algum envolvido na Avaliação Pós-Projeto.

4.1. Missão da INFRAERO

"Atender às necessidades da sociedade relativas à infra-estrutura aeroportuária e aeronáutica, de modo a contribuir para o desenvolvimento sustentável do Brasil, primando pela eficiência, segurança e qualidade".

4.2. Caso INFRAERO-RJ

- Empresa: INFRAERO (Empresa Brasileira de Infra-estrutura Aeroportuária)
- Software utilizado: PSM INSIGHT versão 4.2.1
- Entrevistas realizadas entre abril e junho de 2.007
- Case aplicado e mensurado no segundo semestre de 2005 e primeiro de 2006.

4.2.1. Pontos Principais do Caso

O caso INFRAERO destaca-se pela utilização do PSM INSIGHT com a finalidade de centralizar a mensuração de projetos de departamentos distintos em várias localidades no Brasil. Seu processo de implementação iniciou-se na base INFRAERO-RJ, especificamente no Aeroporto Internacional Tom Jobim (Galeão). Está sendo conduzido por uma grande equipe de projeto, utilizando uma combinação de implementação em fases por módulos e por

[23] Os dados disponíveis sobre a INFRAERO podem ser encontrados no site http://www.infraero.gov.br Acesso em 20 out 2007.

fábricas de software. Outro destaque do caso é a ligação com o Project 2003 Server, o que permitiu grande controle sobre os processos produtivos de vários projetos.

4.2.2. Apresentação da Empresa

A INFRAERO é uma empresa pública com 33 anos de tradição e credibilidade no mercado. Vinculada ao Ministério da Defesa administra 67 aeroportos, 81 unidades de apoio à navegação aérea e 32 terminais de logística de carga. A cada ano, cerca de 330 milhões de pessoas passam por estes aeroportos, sendo aproximadamente 83 milhões de passageiros. Em 2005, o número de operações foi de cerca de dois milhões de pousos e decolagens.

A INFRAERO armazena e paletiza 1,3 milhão de toneladas de cargas aéreas por ano. O Aeroporto Internacional de Guarulhos, em São Paulo, o maior em número de passageiros da rede, gera nada menos do que 53 mil empregos diretos e indiretos.

Tem um movimento diário superior a 100 mil pessoas e um movimento anual que chega a 13 milhões de passageiros. Para gerenciar estas operações com a máxima eficiência, a INFRAERO desenvolveu um plano de obras arrojado, que tem como objetivo modernizar a infra-estrutura aeroportuária brasileira para os próximos dez anos.

É uma empresa nacional acostumada com os diversos sotaques regionais. Administra desde os maiores aeroportos brasileiros até alguns tão pequenos que sequer recebem vôos comerciais regulares – caso de aeroportos cuja função é representar a soberania nacional em áreas longínquas. Além disso, opera aeroportos equipados para funcionar como plataformas de helicópteros e outros cuja vocação está na logística de carga e de manuseio de mercadorias perigosas.

Como empresa pública presente em todo o país, a INFRAERO tem consciência de que todas as suas ações devem ser guiadas pela responsabilidade social.

Sendo assim, implementa e administra ações educativas e culturais voltadas, sobretudo, aos seus funcionários e aos moradores do entorno aeroportuário. A empresa também investe em meio-ambiente, com programas que englobam diversas necessidades ambientais.

4.2.3. A área de Tecnologia de Informação (T.I.)

Depois de pesquisa de mercado e licitação pública, a INFRAERO adotou, na década de 90, a solução de ERP - *Enterprise Resource Planning* mais utilizada para o ambiente mainframe, o *GL Millennium,* da Geac. Em seguida, quando decidiu pela troca do sistema proprietário em favor da arquitetura aberta, a empresa novamente passou a estudar a adoção de uma ferramenta. "Olhamos as opções dos principais competidores, mas o *SmartStream, da Geac*, foi a que mais nos atendia, especialmente pela facilidade de migração e implantação".

Em 2000, a INFRAERO adotou novos módulos do *SmartStream* para viabilizar o seu projeto de expansão e modernização dos aeroportos. Para ampliar seus recursos era fundamental a criação de mecanismos eficientes de controle de caixa. Isto é, receber e controlar tudo que lhe é devido em tempo hábil e, ao mesmo tempo, controlar efetivamente os gastos. Em 2003, a INFRAERO expandiu o *SmartStream* com o módulo *Billing* totalmente customizado para aeroportos.

No ano seguinte, fechou o circuito do ERP com o *Procurement* para controle total dos contratos antes da liberação dos pagamentos. Para o departamento financeiro, sem uma solução de ERP, os processos podem se tornar lentos e burocráticos. Por exemplo, para uma nota fiscal ser paga pelo sistema tradicional recebia até 18 assinaturas. Com o *SmartStream*, isso foi completamente modificado. O processamento é linear, mais simples e totalmente eletrônico. “Conseguimos avançar na produtividade, no controle dos negócios e estamos em dia com os relatórios mensais e informações enviadas aos órgãos controladores".

O *SmartStream* roda dentro do sistema operacional Windows Server e trabalha fortemente integrado com a base de dados da Microsoft, o SQL. O sistema é acessado por cerca de 2000 usuários lotados nos 67 aeroportos e nas 82 unidades de apoio à navegação aérea da INFRAERO com acesso ao DAC - Departamento de Aviação Civil, órgão normatizador do Sistema de Aviação Civil Brasileira, e salas AIS - Serviço de Informações Aeronáuticas.

4.2.3.1. Sobre a Geac

Geac (TSX: GAC e NASDAQ: GEAC) é uma companhia global de software que atende às necessidades dos *Chief Financial Office*. Geac tem o melhor em tecnologia de produtos e serviços para auxiliar nas questões de negócio que o *Chief Financial Office*

enfrenta para fazer mais com menos em um ambiente crescentemente competitivo e sob pressão.

A INFRAERO é a primeira empresa brasileira a receber o Prêmio Mundial de Excelência no uso do *SmartStream* - software de gestão financeira da Geac, empresa global de software de *Business Performance Management*. Este reconhecimento foi feito durante o evento anual da companhia, o Geac Alliance 2004, realizado em Chicago, EUA.

O Geac Aliance 2004 contou com a participação de aproximadamente 1000 clientes mundiais. Durante o evento, foram apresentadas as tendências futuras em Tecnologia da Informação e *best practices* de Finanças e no gerenciamento da implantação de ERP - *Enterprise Resource Planning*. Todos os anos, a Geac elege os melhores cases de cada categoria de produto.

Em 2005, com um fluxo de passageiros acima dos 30 milhões por ano, os aeroportos da Guarulhos, Congonhas e Viracopos passaram por uma série de reformas e ampliações para atender à crescente demanda de todo o complexo aeroportuário, que compreende desde os passageiros, às companhias aéreas e órgãos do governo (Polícia Federal, Vigilância Sanitária, dentre outras). As reformas incluem novos terminais de cargas, novas salas e pontes de embarque/desembarque, que foram instaladas nos três aeroportos. Para acompanhar as ampliações estruturais, a INFRAERO investiu na modernização de toda a *rede de dados e na implantação de Telefonia IP* nos aeroportos de Guarulhos, Congonhas e Viracopos. Para isto, optou pela 3Com, em parceria com a integradora Ziva, como provedora deste novo sistema de dados e voz. Os três aeroportos atendidos no projeto são, em números de passageiros e cargas, os maiores do país e, no caso de Guarulhos, da América Latina. Em Guarulhos, são mais de 15 milhões de passageiros/ano e, em Congonhas, o aeroporto de maior movimento da América do Sul, são 12 milhões de passageiros/ano. Já o aeroporto de Viracopos desponta para o segmento de carga aérea internacional, com um volume de cerca de 170 mil toneladas de carga aérea/ano.

Com um complexo tão gigantesco, a INFRAERO precisava, urgentemente, de uma modernização em sua rede de dados, não apenas para atender as ampliações dos terminais, mas para adequar sua tecnologia às centenas de empresas que atende em todo o complexo aeroportuário, já que a sua rede é disponibilizada a todas estas instituições. A antiga infra-estrutura de TI – formada por redes ATM 155 Mbps (core) e 10/100 Mbps (borda) – precisava ser atualizada, acabando com inúmeros problemas, como paradas no sistema, dificuldade para reposição de peças, altos custos com manutenção e o pior, a baixa credibilidade.

Manter o antigo sistema era um risco muito grande à INFRAERO e a todas as empresas que operam nos aeroportos, pois, entre outros problemas, poderia ocasionar atrasos em vôos de passageiros e de cargas, trazendo inúmeros prejuízos como, por exemplo, perda de cargas perecíveis ou multas por atrasos na entregas de encomendas. De acordo com o Major Bernardo Levino dos Santos, gerente de tecnologia da informação da INFRAERO, o ponto crítico ocorreu quando uma empresa aérea não pôde participar da rede da INFRAERO porque o tráfego com o qual trabalhava era *Gigabit*, não compatível com o *Fast Ethernet* utilizado pela instituição.

Comenta o Major (2007):

> Como um centro de excelência, precisamos oferecer, acima de tudo, credibilidade aos que trabalham conosco e a nossos clientes [...] Hoje, estamos com um ambicioso plano de TI para caminharmos com a velocidade do mercado e a 3Com, desde 98, vem nos auxiliando neste processo.

Para atender os 10 mil pontos físicos – total de pontos das redes de Guarulhos, Congonhas e Viracopos – foram utilizados os *Switches 7700* da 3Com para o core da rede. São *Switches Gigabit Ethernet*, com portas 10/100 de alto desempenho. Este *Switch* foi eleito pelo centro de pesquisas independente *Tolly Group* como o "único dispositivo testado com perda zero de desempenho em todos os tamanhos de pacote com os quais foi testado". Os *switches 7700* adquiridos vão permitir, ainda, uma futura expansão para um *backbone* 10Gigabit, quando a demanda de tráfego crescer. Para a borda das redes, foram utilizados os *switches 4400* da 3Com.

Além dos *switches* de rede, foram instalados Pontos de Acesso *wireless* 802.11 da 3Com nos terminais de cargas dos aeroportos de Guarulhos (200 pontos) e Viracopos (120 pontos), para oferecer dinâmica de trabalho e mobilidade aos funcionários. Os produtos possuem tecnologia *Spread Spectrum*, técnica de espalhamento espectral com sinais de rádio freqüência de banda larga, provendo maior segurança, integridade e confiabilidade, além de maior consumo de banda.

Para o sistema de telefonia IP, foi utilizado o sistema NBX100 da 3Com nos aeroportos de Guarulhos e Viracopos. O sistema está em fase inicial de implantação em Guarulhos e em Viracopos. O NBX100 da 3Com oferece as funcionalidades e benefícios de um PABX IP, como gravação de voz, tarifação e bilhetagem, call center, voice mail, assistência de chamadas e auto-atendimento (URA). A solução ainda apresenta recursos de conferência, *software call assistance* (integração do ramal telefônico do usuário com o PC) e administração via Web, o que facilita sua instalação e administração. Segundo o gerente de

contas da 3Com, Marcelo Naves, outro ponto a se destacar na solução implementada foi o aumento do nível de controle de uso e segurança na rede. “Isto foi possível graças à utilização dos mais avançados protocolos de controle de acesso, como 802.1x e ACLs (Listas de Controle de Acesso) em todos os pontos da rede”, afirma.

“A solução 3Com permite uma segmentação completa do tráfego, separando dados operacionais, administrativos e dos concessionários, além de bloquear de forma automática tráfegos indesejados ou nocivos à rede.” Além dos benefícios já citados, a INFRAERO ainda pôde perceber inúmeros outros, como aumento da disponibilidade do sistema, redução de despesas com manutenção e reposição de peças, aumento da QoS (Qualidade de Serviço) e maior agilidade nos serviços prestados pelas companhias aéreas e órgãos públicos. Os passageiros ganharam serviços mais seguros, precisos e ágeis.

"Os investimentos já somam 7 milhões de reais. Além da rede Gigabit, estão sendo instalados *access points wireless*, para acesso sem fio, e telefonia IP" (NAVES, 2007)
Entre os equipamentos da nova rede, está uma central telefônica IP para até 1500 dispositivos, além de 30 telefones IP e 20 *access points wireless lan*. A rede suporta 1400 funcionários da INFRAERO, que utilizam mais de 750 micros e 500 impressoras, além de fornecer dados para o público e para as companhias aéreas por meio de sistemas como o SIV (Sistema Informativo de Vôo), que mostra informações sobre chegadas e partidas dos cerca de 400 vôos diários.

A administração de um aeroporto é bastante complexa e envolve diversos sistemas. "Temos um sistema chamado Sara (Sistema de Alocação de Recursos Aeroportuários), que é alimentado com informações sobre o tipo de aeronave que estará chegando, o número de passageiros, o volume de bagagem e outros dados. Com base nisso, o avião é alocado em um ou outro terminal". Após a aterrissagem, toda uma logística entra em operação e uma rede bem estruturada é fundamental.

4.2.4. Infra-estrutura de TI

- SERVIDORES: 60
- PLATAFORMA: Windows 2000 e Windows XP
- BANCO DE DADOS: Oracle 8i (Update para 11g Oracle) e SQL Server
- STORAGE: Tandberg com 750 GB
- REDE: Gigabit Ethernet da 3Com
- FUNCIONÁRIO DE TI: 92 profissionais

4.2.5. Histórico de Implementação

Segundo o gerente de Tecnologia da Informação, entre os próximos projetos de TI da INFRAERO, dois são destaques: o *outsourcing* de impressão, cuja licitação deverá sair ainda neste ano e a construção de duas salas-cofres, em Guarulhos e em Congonhas. A Regional Sudeste da INFRAERO planeja para 2008 a construção das salas onde ficarão os servidores, backups e storage. Nacionalmente, a INFRAERO também pretende elaborar e implantar uma política padronizada de segurança em TI .

Além das modernizações efetuadas acima, a INFRAERO esta disponibilizando na internet o SIV - Sistema de Informação de Vôo que são as televisões que estão no saguão do aeroporto e informam partidas e chegadas, onde podemos visualizar agora via internet em tempo real o movimento de vôos de vários aeroportos administrados pela INFRAERO , já que hoje cada aeroporto tem seu próprio sistema. Outro sistema que desenvolvemos em parceria com a G&P projetos e sistemas, no Rio de Janeiro, foi o Internet Banking INFRAERO (IBI).

O processo de implementação iniciou em março de 2006 na INFRAERO-RJ, base Aeroporto Tom Jobim e durou até janeiro de 2007, ou seja, a fase de testes e manutenabilidade do produto. Na verdade neste projeto foram alteradas todas as *front-ends* e permitido o acesso via internet em tempo real de tais informações, pois um sistema desktop parecido já existia.

Como citado, um dos requisitos da INFRAERO na escolha da solução era que cada aeroporto desenvolvesse a sua solução dentro de um contexto previamente determinado por Brasília, permitindo que cada aeroporto desenvolvesse a sua interação com os dados em uma única Base de dados.

4.2.6. Implementação: problemas

Em qualquer ambiente de trabalho há situações de estresse e problemas de relacionamento pessoal. Em vários projetos tínhamos a certeza de problemas difíceis que afetariam os nossos funcionários e suas tarefas. Os processos de gerência abrangem todos os aspectos de construção do produto. Precisamos intensivamente de ferramentas como sistemas para controle de versão e linguagens, técnicas organizacionais e de administração de pessoas, ferramenta para acompanhamento do projeto e medição da qualidade. Algo, que

proporcionasse, através de um conjunto de dados, medidas de extrema ajuda para auxiliar a tomada de decisões.

O aumento da eficiência tornou-se uma condição imprescindível para garantir a sobrevivência. Dessa forma foi preciso desenvolver mecanismos diferentes e a resposta veio da implementação de um software de mensuração das práticas implementadas no projeto. Em programas de computador, o problema de complexidade e tamanho é ainda mais grave, em razão das interações entre os diversos componentes do sistema, precisávamos de uma ferramenta que nos auxiliasse a solucionar questões envolvidas na construção de um projeto de software?

- Cronogramas não observados
- Módulos que não operam corretamente quando combinados
- Programas tão difíceis de usar que são descartados
- Programas que simplesmente param de funcionar

E quando estamos tratando de software essa zona de sombra que envolve fatores desconhecidos é bem mais abrangente. A volatilidade dos requisitos é um das maiores causas de insucesso de projetos de software. Sem uma definição precisa daquilo que se pretende construir, perde-se tempo, mais erros são cometidos e a qualidade do produto final é incerta. Um dos problemas mais críticos que enfrentamos estava focado na compreensão do analista de sistemas ao produto a ser desenvolvido, este, descrevia algo simplesmente técnico, algo relacionado ao seu mundo.

Um software não é um simples conjunto de funções desconexas. As operações e dados estão relacionados, havendo encadeamento entre si. Os analistas intervêm com clientes, para organizar as operações de maneira que se tornem eficientes para a realização da tarefa e fáceis de usar, faltava algo que acompanhasse exatamente os nossos analistas.

O gerente do projeto pode incluir preocupações com respeito aos recursos envolvidos. Limitações como cronograma ou orçamento, por exemplo, podem levar a um estudo de possíveis simplificações a serem discutidas com o cliente.

A implementação provavelmente envolverá o uso de tecnologia adequada, como linguagens, hardware e sistema operacional. Se a equipe de desenvolvimento não está bem familiarizada com as ferramentas, é possível que exista uma certa pressão para simplificar o software e reduzir sua funcionalidade.

4.2.7. A descoberta

A administração de um projeto envolve o uso correto de informação: o administrador trabalha cercado de especificações de projeto, diagramas, orçamentos e cronogramas. Chegamos à conclusão que as métricas fornecem uma parte importante dos dados necessários para administração de um projeto de software.

Os números têm um caráter particular: permitem análises, comparações e combinações que são impossíveis de fazer com outros tipos de informação (produtividade de programadores, horas de trabalho e evolução de cronograma ganham enorme importância).

É vital perceber que se deve administrar usando os números, sem ser administrado por eles. No domínio das ciências exatas e da engenharia é diferente, um pequeno erro de medida pode ter conseqüências bastante significativas.

Embora relativamente novo, o PSM entra em cena munido de valiosíssimas credenciais. O modelo foi elaborado e vem sendo atualizado por renomados profissionais da área de software *Process Improvement*. Além disso, o PSM foi utilizado como base para a elaboração da *Process Área Measurement and Analysis* do CMMi – o Modelo integrado que poderá substituir o CMM no futuro e esta sendo o nossa proposta atualmente na INFRAERO-RJ.

Algumas organizações que participam ativamente do desenvolvimento do PSM e que nos levaram a adotá-lo são: Força Aérea, Exército e Marinha Norte-Americanos e a Boeing dentre outras.

4.2.8. A melhoria de processos de desenvolvimento de software

Adotamos o PSM na INFRAERO-RJ, no desenvolvimento de projetos de software, no segundo semestre de 2005. Foram utilizados conjunto de métodos, formulários e scripts (roteiros) para planejar, medir e gerenciar o trabalho dos indivíduos. O Objetivo geral do PSM é a produção de produtos de software sem defeitos, respeitando prazos e custos planejados.

Um princípio importante do PSM é lembrar que cada indivíduo é diferente e, com isso, o planejamento do trabalho deve ser baseado em dados com o desempenho individual. Por exemplo, a tarefa de um gerente ao definir o cronograma de uma equipe pode se basear em índices mínimos e máximos de produtividade de cada desenvolvedor.

De um ponto de vista prático o PSM procura resolver dois problemas:

1- Especificar as medidas a serem utilizadas

2- O modelo de medição

Para cada projeto de nossa unidade organizacional definimos:

- *Atributos:* Horas trabalhadas no projeto (*timesheet*s), funcionalidade oferecida pelo projeto sempre sofrendo atualizações.

- *Métodos*: Contar horas (obtendo o esforço do projeto), contar pontos de função (obtendo o tamanho funcional do Projeto).

- *Medidas básicas*: esforço do projeto em horas, tamanho do projeto em pontos de função.

- *Função:* Dividir tamanho do projeto pelo esforço (obtendo Pontos de função/Hora).

4.3 Case: a aplicação do PSM

O modelo aplicado com a ferramenta PSM INSIGHT é o case *Internet Banking INFRAERO-RJ*, uma parceria com a principal instituição financeira e as companhias aéreas.

Seguem algumas características do projeto utilizado.

- ✓ Nome do Projeto : Internet Banking INFRAERO
- ✓ Objetivo: Criar uma solução de integração Bancária que ofereça aos clientes "companhias aéreas" os serviços de: saldo, saque, depósito, transferências entre contas e bancos.
- ✓ Patrocinador: INFRAERO-RJ
- ✓ Motivo da Solicitação: Automatização das transações financeiras entre clientes.
- ✓ Período: Abril 2005 a Novembro 2006
- ✓ Equipe: Profissionais G&P, Gestor de Projetos G&P, Supervisor de Sistemas.

4.3.1 Estabelecer e sustentar comprometimento

O primeiro passo da aplicação do PSM é garantir a estrutura organizacional para o projeto. Esta estrutura é composta de três tarefas:

- Obter apoio dos patrocinadores.

- Definir as responsabilidades de cada membro da equipe.

- Verificar recursos disponíveis.

4.3.2. Planejar a Mensuração

A atividade de planejar a mensuração utiliza a ferramenta PSM INSIGHT para sua aplicação. Os passos abaixo descrevem os processos que foram efetuados.

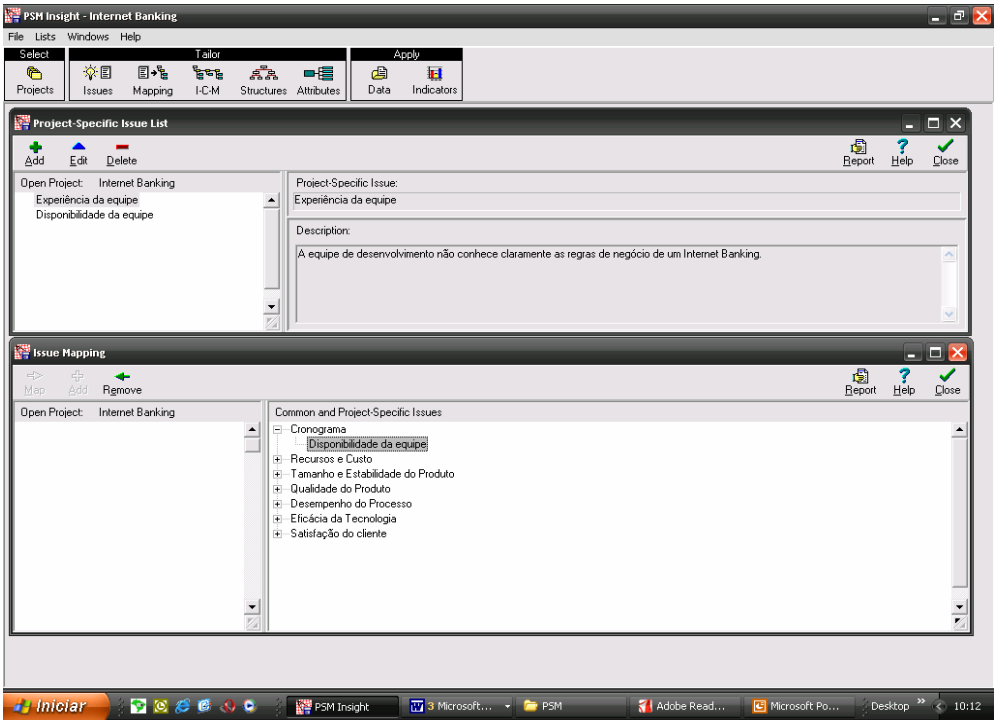
1° Cadastro do Projeto no PSM INSIGHT.

No menu Projects cria-se um novo projeto (ADD), informando seu nome e suas características principais.

2° Cadastro de Issues (Pontos Críticos) e mapeamento

A primeira tarefa da fase planejar mensuração é a identificação de pontos críticos específicos do projeto.

O cadastro desses pontos críticos é efetuado através do menu Issues. Após cadastrados se faz necessário mapear os pontos críticos específicos dentro dos sete pontos críticos comuns do PSM; isto é feito através do menu *mapping*. (Figura 29)



Figuras 28 / 29 - Cadastro de Issues/mapping
Fonte: PSM INSIGHT - INFRAERO

3° Cadastro de Estruturas de agregação

O menu *structures* é responsável pelas três estruturas de agregação do projeto. É necessário que as três estruturas sejam cadastradas, pois elas serão utilizadas pelas métricas.

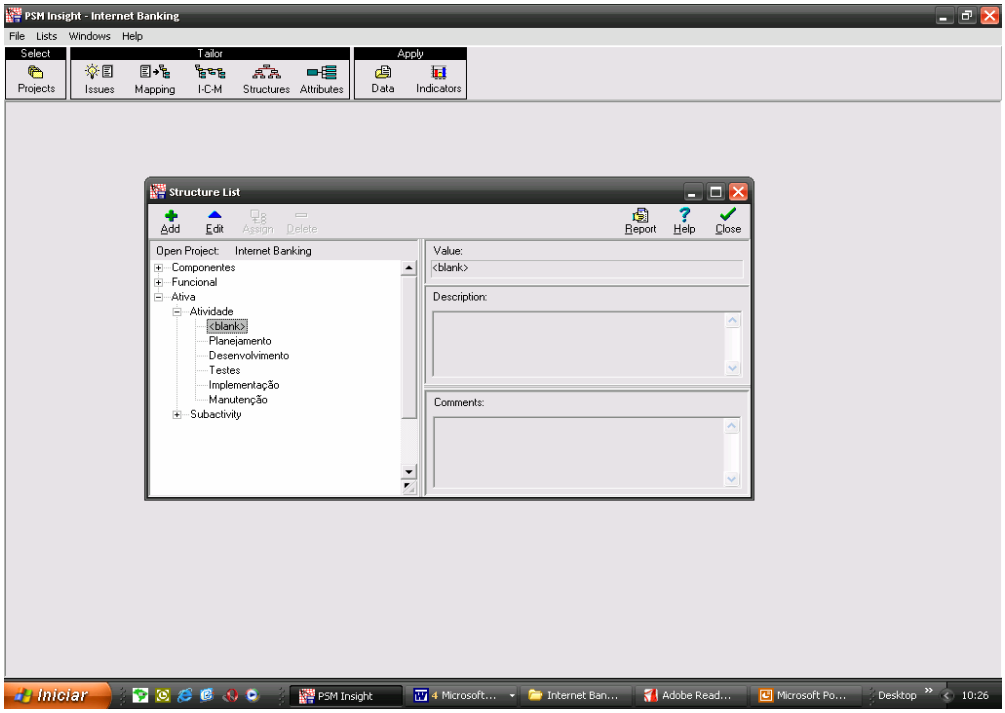


Figura 30 - Cadastro de estruturas de agregação
Fonte: PSM INSIGHT - INFRAERO

4° Cadastro de Atributos

O menu *attributes* é responsável pelo cadastro de atributos do projeto. O PSM traz como padrão mais de 100 atributos que já podem ser utilizados.

Para o cadastro deve-se selecionar o atributo desejado na lista e clicar na opção *Assign*; é apresentada a tela *Attribute setup*; deve-se clicar em ADD para adicionar os valores para o atributo.

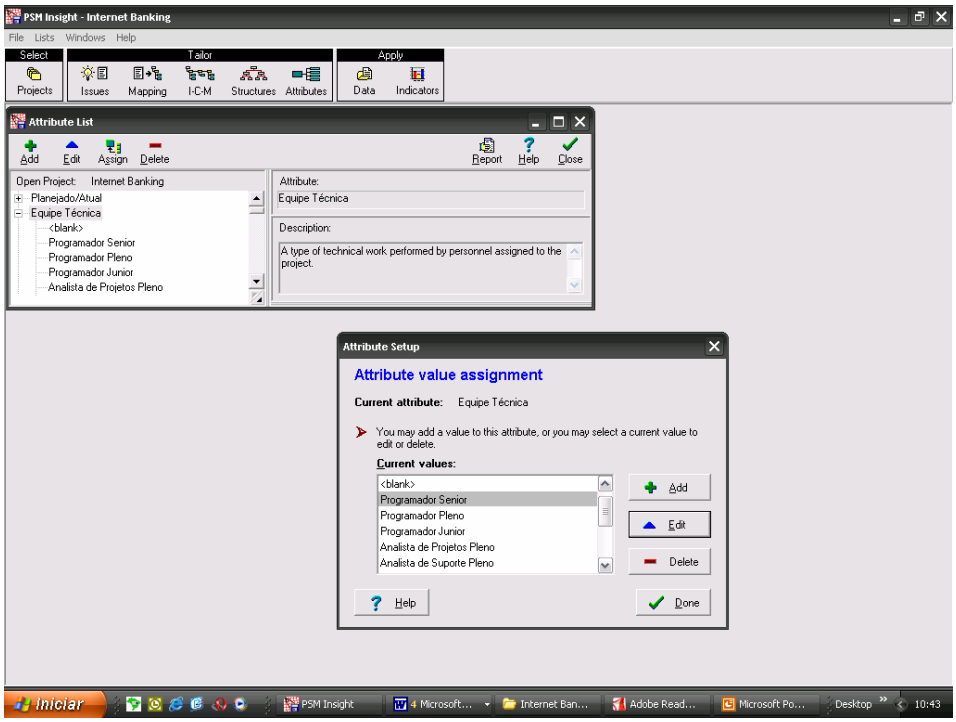


Figura 31- Cadastro de atributos
Fonte: PSM INSIGHT - INFRAERO

5° I-C-M (Pontos críticos, categorias, métricas)

Este menu é responsável pela seleção de métricas que são aplicadas no projeto. A Figura 32 demonstra como sua estrutura esta organizada.

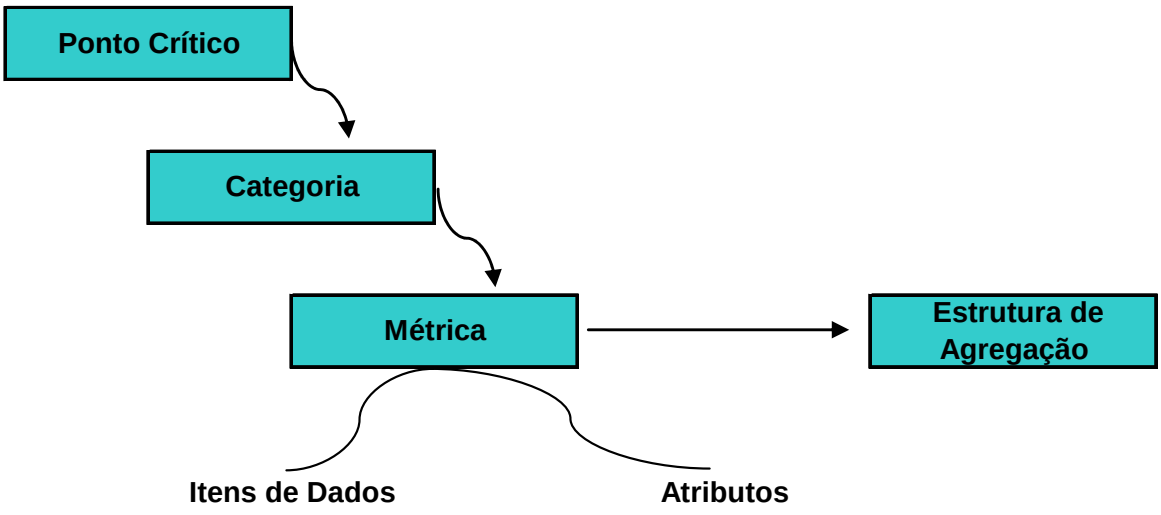


Figura 32 - Estrutura do I-C-M
Fonte: PSM INSIGHT - INFRAERO

✓ Ponto Crítico: Selecionar um dos sete pontos críticos comuns do PSM. Exemplo: Recursos e Custo.

✓ Categoria: Cadastrar o item a ser medido.

Exemplo: Equipe, tamanho do produto.

O PSM já traz algumas categorias comuns.

✓ Métrica: Cadastrar as métricas que serão utilizadas para medir a categoria. Exemplo: Esforço, linhas de código.

O PSM já traz algumas métricas cadastradas.

✓ Estrutura de agregação: Informar a estrutura de agregação que está sendo utilizada para o cálculo da métrica.

✓ Atributos: Selecionar os atributos que deverão fazer parte desta métrica. Exemplo: Planejamento, Equipe técnica.

✓ Itens de dados: São as informações, números que alimentarão as métricas. Exemplo: Horas trabalhadas, número de linhas de código.

Para o cadastro do I-C-M no PSM INSIGHT:

✓ Selecionar o ponto crítico.

✓ Cadastrar uma nova categoria que será medida (Figura 33)

✓ Cadastrar as métricas que serão utilizadas na categoria (No cadastro de métricas que serão especificados os itens de estrutura de agregação, atributos e itens de dados.) (Nas Figuras 34 e 35)

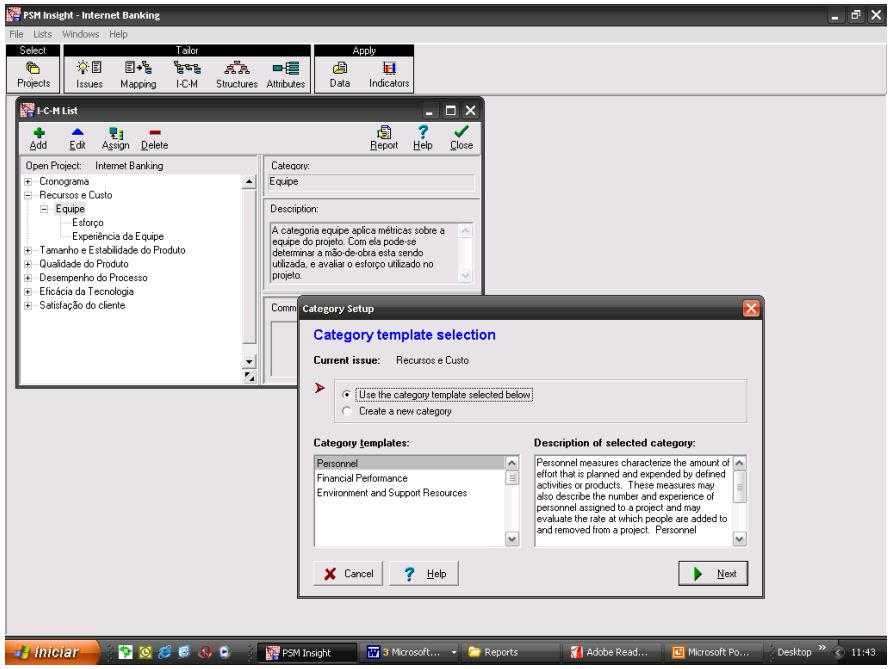


Figura 33 - Cadastro de Categorias
Fonte: PSM INSIGHT - INFRAERO

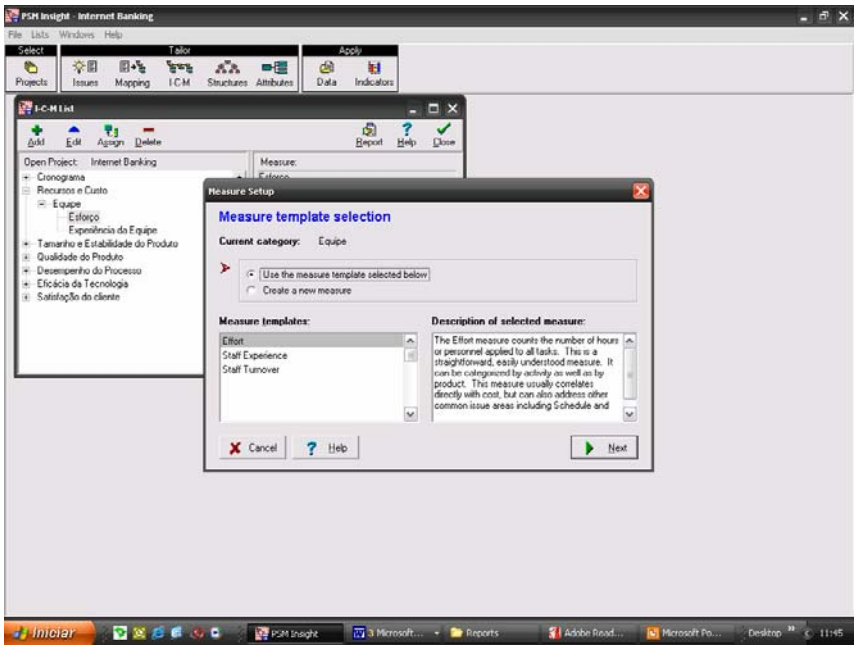


Figura 34 - Cadastro de Medidas
Fonte: PSM INSIGHT - INFRAERO

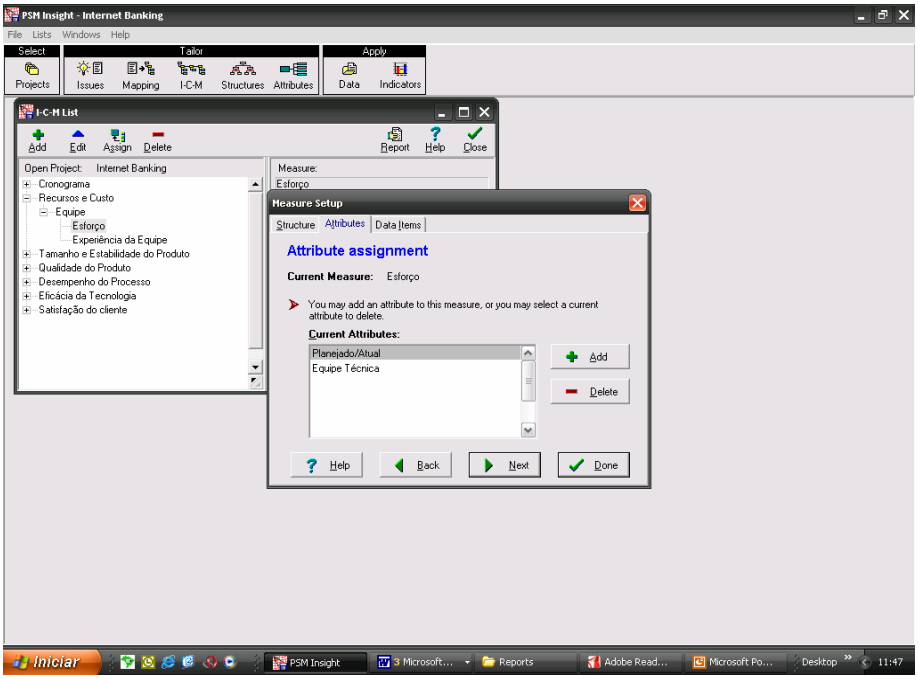


Figura 35 - Setup Medida
Fonte: PSM INSIGHT - INFRAERO

4.3.3. Executar a Mensuração

Agora que já estão cadastrados no PSM INSIGHT os pontos críticos, as categorias que serão medidas e as métricas, o próximo passo é executar a mensuração. É importante seguir os procedimentos de aplicação da métrica para que seus resultados retratem a realidade.

Esta tarefa no PSM INSIGHT esta dividida nos menus Data e Indicators

1° Inserir resultados das métricas.

A inserção dos resultados obtidos, no PSM INSIGHT pode ocorrer de duas maneiras :

- Manual: Os dados deverão ser inseridos manualmente pelo menu data. Para isto deve-se selecionar a métrica desejada e clicar em open. Será aberta uma base para a inserção dos resultados obtidos com a métrica. (Figura 36)

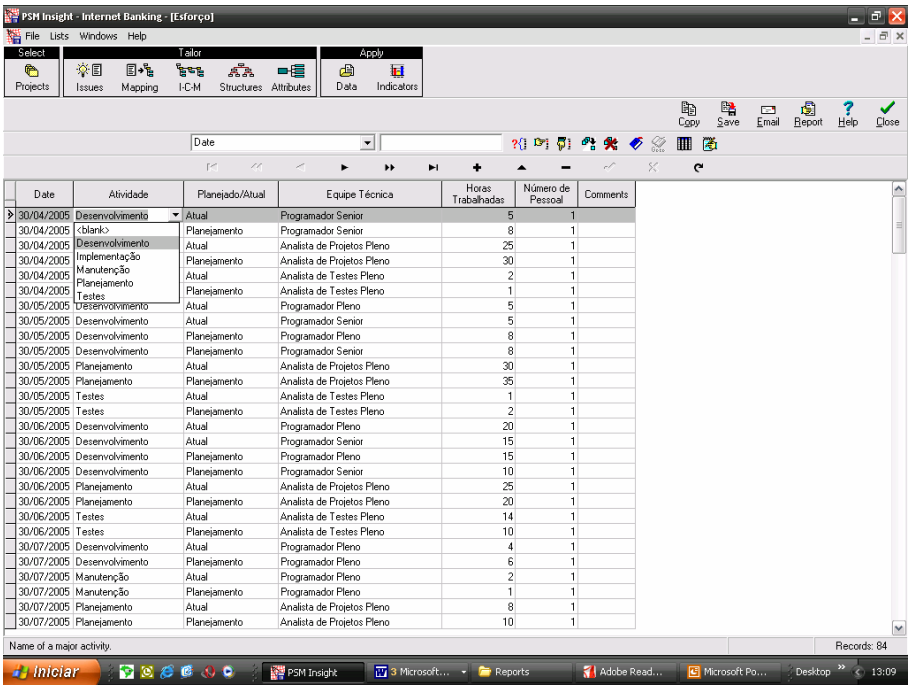


Figura 36 - Inserção de Dados - Manual
Fonte: PSM INSIGHT - INFRAERO

- Automática: O PSM INSIGHT possui uma ferramenta de importação de dados que funciona diretamente com arquivos texto e banco de dados. Esta maneira de inserção de dados é recomendável, pois algumas métricas necessitam de vários registros, o que impossibilita a inserção manual. Esta ferramenta é um software separado denominado de PSM INSIGHT *Import/Export* (Figura 37).

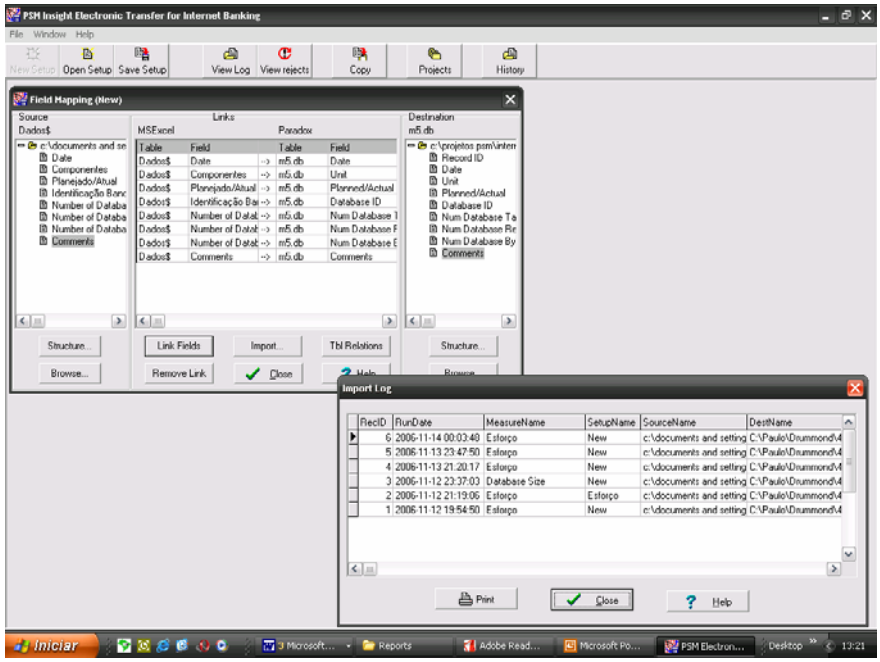


Figura 37 - Inserção de Dados – PSM Import/Export
Fonte: PSM INSIGHT - INFRAERO

2° Indicadores.

Com os resultados já inseridos no PSM INSIGHT, o menu *Indicators* é responsável pela criação de gráficos indicadores que vão auxiliar a análise dos resultados obtidos pelas métricas.

Para criar os indicadores, basta selecionar o ponto critico desejado e clicar em *add*, será apresentado um setup para a seleção da métrica e configurações do gráfico. (Figura 38).

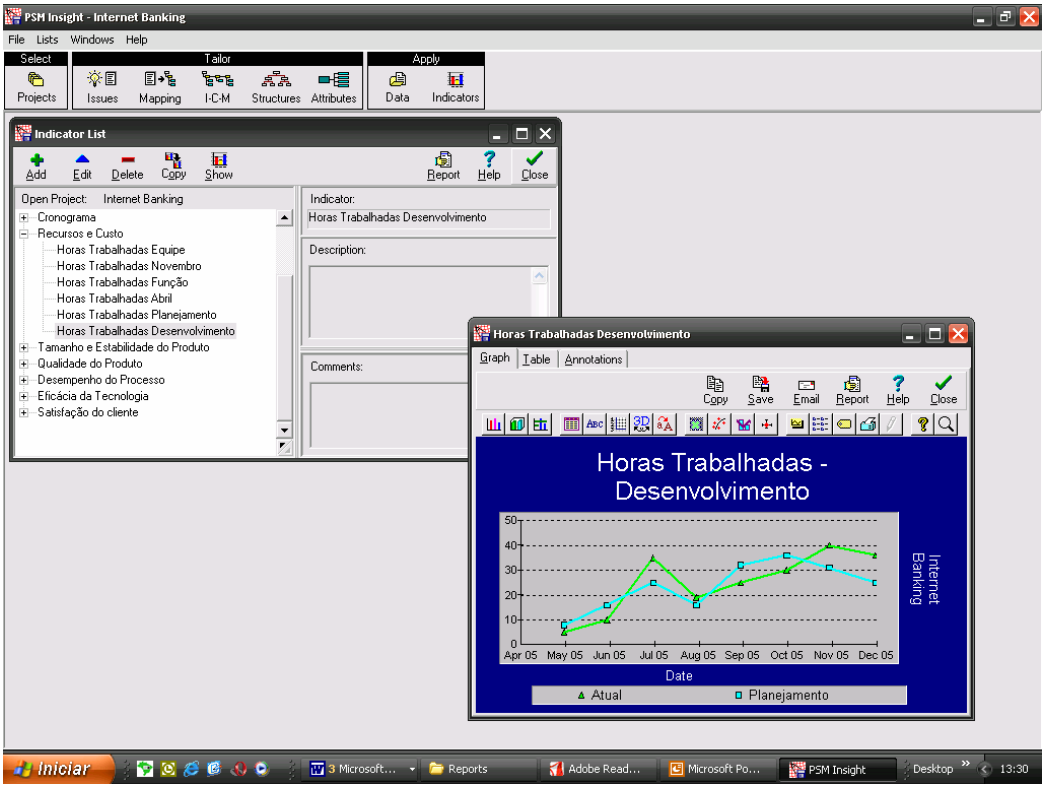


Figura 38 - Indicadores
Fonte: PSM INSIGHT - INFRAERO

4.3.4. Avaliar Mensuração

A fase de avaliar mensuração é a última fase do PSM. Nela é necessário reavaliar todo o processo de planejamento e execução das métricas e documentá-lo para projetos futuros. Também é nesta fase que uma auditoria externa se faz necessária para validar o processo e verificar se as normas do ISO/IEC 15939 estão sendo cumpridas.

4.4. Plano de Mensuração – Projeto Internet Banking

Até este momento foi apresentado, neste capítulo, os passos para a aplicação do PSM em um projeto. Agora apresentamos o plano de mensuração criado para o projeto Internet Banking na INFRAERO-RJ.

4.4.1. Plano de Mensuração – Projeto Internet Banking INFRAERO-RJ

Parte 1 – Introdução

O objetivo deste projeto é desenvolver uma solução de integração bancária que ofereça aos clientes os serviços: saldo; saque; deposito; transferências entre contas e bancos.

Parte 2 – Descrição do projeto

O projeto deve atender as seguintes requisições técnicas:

- Segurança:

- Implementar a tecnologia *Captcha*.
- Criar *triggers*[24] para gerar *logs* das operações efetuadas.

- Confiabilidade:

- Implementar autômatos para validação de dados.

- Transferência bancaria entre contas e outros bancos. Implementar a tecnologia to *phase commit*[25] para controlar as transações.

Parte 3 – Regras de mensuração, responsabilidades, comunicação

- Regras de mensuração

Todos os processos que envolverem o projeto Internet Banking devem ser quantificados.

Todos os envolvidos no projeto, ao final da jornada de trabalho devem informar dados para que possam ser utilizados em métricas (Dados como horas trabalhadas, problemas

[24] Gatilho: Função utilizada para executar uma instrução diretamente no banco de dados.
[25] Protocolo de envio e confirmação de informações.

encontrados, componentes desenvolvidos). A informação desses dados, na maioria das vezes, é automática, através da coleta de dados direta. Podem existir dados que a informação é manual, neste caso o membro da equipe deve informar o analista de métricas.

- Responsabilidades

As responsabilidades de cada membro do projeto são:

- Envolvidos com o projeto: Informar os dados corretamente. Não ocultar informações.

- Analista de Métricas: Avaliar os dados recebidos da equipe, trabalhar os dados no PSM , gerar relatórios de controle.

- Gerente de projeto: Analisar os resultados dos indicadores e tomar decisões para a melhora do projeto.

- Comunicação

A comunicação deve ocorrer através de relatórios periódicos ou reuniões de alinhamento, que informarão a equipe como está o desempenho do projeto e quais alterações que o projeto poderá sofrer.

Cria-se um banco de dados com todos os resultados obtidos. Este banco ficará disponível para equipe consultar dados para projetos futuros que necessitem da análise de um portfolio.

Parte 4 – Descrição das informações do projeto

O projeto é uma demanda que surgiu devido a necessidade de integração do software de gestão empresarial da INFRAERO e as companhias aéreas agregadas. O prazo total para o desenvolvimento do software é de dezenove meses, com início em Abril/2005 e término previsto para novembro/2006.

São consideradas informações prioritárias do projeto:

- A equipe não tem experiência na área do projeto (bancária).

- As tecnologias *Captcha, to phase commit*, necessitam de pesquisas e treinamentos para sua aplicação.

- O tempo da equipe é limitado.

A equipe responsável pelo desenvolvimento é composta por quatro profissionais, Analistas de Sistemas e programadores, somando os conhecimentos de: programação,

engenharia de software, suporte e softwares de Gestão Empresarial.

Parte 5 –Especificação das Métricas

As métricas escolhidas para o acompanhamento deste projeto foram:

Nome	Complexidade Ciclomatica - tcta
Ponto Crítico	Qualidade do Produto
Categoria da Métrica	Suporte e manutenibilidade
Itens de Dados	Números de pontos de decisão, Nível de complexidade
Atributos	Planejamento/Atual
Estrutura de Agregação	Componentes
Tipo de dado	Número
Critérios	Esta medida responde perguntas como : - Quantos componentes complexos existem neste projeto? - Que componentes são os mais complexos? - Que componentes deveriam estar sujeito a revisão?
Freqüência de aplicação	Semanalmente
Indicadores	Evolução da complexidade do software
Nome	Componentes
Ponto Crítico	Tamanho e Estabilidade do Produto
Categoria da Métrica	Tamanho físico e estabilidade
Itens de Dados	Número de unidades, número de unidades adicionadas, número de unidades deletadas, número de unidades modificadas
Atributos	Planejamento/Atual, Linguagem, Status de entrega
Estrutura de Agregação	Componentes
Tipo de dado	Número
Critérios	Esta medida responde perguntas como : - Quantos componentes precisam ser implementados e testados ? - Quanto mudou a linha de base de sistema aprovada?
Freqüência de aplicação	Quinzenalmente
Indicadores	Evolução do número de componentes.
Nome	Datas
Ponto Crítico	Cronograma
Categoria da Métrica	Planejamento
Itens de Dados	Data Início, Data Termino
Atributos	Planejamento/Atual, Eventos
Estrutura de Agregação	Componentes
Tipo de dado	Data
Critérios	Esta medida responde perguntas como : - O programa atual é realístico? - Quantas atividades estão sendo efetuadas simultaneamente ? - Com que freqüência mudou o programa? - A data de cumprimento projetada para o projeto será cumprida? - Que atividades, eventos, ou produtos estão no prazo, adiantado, ou atrasado?.

Freqüência de aplicação	Planejamento de cronograma. È aplicado quanto necessário. Sua atualização pode até ser diária.
Indicadores	Indicador de GANTT.
Nome	Esforço
Ponto Crítico	Recursos e custos
Categoria da Métrica	Equipe
Itens de Dados	Horas trabalhadas, Número de pessoal
Atributos	Planejamento/Atual, Equipe Técnica
Estrutura de Agregação	Ativa
Tipo de dado	Horas, Número
Critérios	Esta medida responde perguntas como: - Recursos de desenvolvimento estão sendo aplicados de acordo com plano? - Certas tarefas ou atividades estão com os valores de esforço acima ou abaixo do planejado ?
Freqüência de aplicação	Diária
Indicadores	Horas trabalhadas equipe Horas trabalhadas por função Horas trabalhas Abril Horas trabalhadas Novembro Horas trabalhadas planejamento Horas trabalhadas desenvolvimento
Nome	Experiência da Equipe
Ponto Crítico	Recursos e custos
Categoria da Métrica	Equipe
Itens de Dados	Número de pessoal, anos de experiência
Atributos	Planejamento/Atual, fator de experiência
Estrutura de Agregação	Ativa
Tipo de dado	Número
Critérios	Esta medida responde perguntas como - Pessoal experiente suficiente está disponível? - Formação adicional será necessária?
Freqüência de aplicação	Trimestral
Indicadores	Não há
Nome	Linhas de Código
Ponto Crítico	Tamanho e Estabilidade do Produto
Categoria da Métrica	Tamanho físico e estabilidade
Itens de Dados	Número de linhas de código, número de linhas de código
	adicionadas, número de linhas de código deletadas, número de linhas de código alteradas.
Atributos	Planejamento/Atual, Linguagem, Status de entrega, Eventos, Tecnologia reutilizada
Estrutura de Agregação	Componentes
Tipo de dado	Número

Critérios	Esta medida responde perguntas como : - Como preciso o orçamento de tamanho de software em o qual era o programa e planos de esforço eram baseados? - Quanto mudou o tamanho de software? Em que componentes aconteceram mudanças? - Qual é o tamanho alocado a cada alteração?
Freqüência de aplicação	Diária
Indicadores	LOC – Componentes
Nome	Tamanho do Banco de Dados
Ponto Crítico	Tamanho e Estabilidade do Produto
Categoria da Métrica	Tamanho físico e estabilidade
Itens de Dados	Número de tabelas, número de gravações, tamanho da base
Atributos	Planejamento/Atual, Identificação do banco de dados
Estrutura de Agregação	Componentes
Tipo de dado	Número
Critérios	Esta Medida Responde Perguntas Como - Quanto dados tem que ser controlado pelo sistema? - Quantos tipos de dados diferentes têm que ser endereçados? - Como esta o espaço de armazenamento do banco ?
Freqüência de aplicação	Diária
Indicadores	Registros no banco de dados

Quadro 17 - Parte 5 – Plano de mensuração – Classificação das medidas
Fonte: PSM INSIGHT – INFRAERO

Parte 6 – Estrutura de agregação do projeto

 Componentes:

Saque
Deposito
Agenda_Pagamento
Agenda_Transferência
Transferência
Cliente
Uf
DAO
JSP
To phase Commit
Cliente_Telefone
Tipo_Telefone
Tipo_Conta_Corrente
Conta_Corrente
Agencia
Cidade
Autômatos
SQL
Captcha

 . Funcional: Serviços de Sistema
Interfaces
Banco de Dados
Regras de negócio
 Ativa: Planejamento Desenvolvimento Testes Implementação Manutenção

Parte 7 – Indicadores

Os indicadores criados para o acompanhamento desse projeto foram:

Nome	Horas Trabalhadas Equipe
Ponto Crítico	Recursos e Custo
Medidas Utilizadas	Esforço
Decisões	Avaliar como esta o esforço atual com o esforço planejado.
Nome	Horas Trabalhadas Abril / Novembro
Ponto Crítico	Recursos e Custo
Medidas Utilizadas	Esforço
Decisões	Avaliar a distribuição do esforço por atividades durante o mês.
Nome	Horas Trabalhadas Função
Ponto Crítico	Recursos e Custo
Medidas Utilizadas	Esforço
Decisões	Avaliar como esta à divisão de esforço entre as funções
Nome	Horas Trabalhadas Planejamento / Desenvolvimento
Ponto Crítico	Recursos e Custo
Medidas Utilizadas	Esforço
Decisões	Avaliar o desempenho de uma tarefa durante o ciclo do projeto.
Nome	LOC – Classes
Ponto Crítico	Tamanho e estabilidade do produto
Medidas Utilizadas	Linhas de código
Decisões	Avaliar o tamanho de um componente.
Nome	Registro no banco de dados
Ponto Crítico	Tamanho e estabilidade do produto
Medidas Utilizadas	Tamanho do banco de dados
Decisões	Avaliar o crescimento de um banco de dados. Informação útil para
	determinar armazenamento de disco e controle de performance do banco de dados.
Nome	Número de componentes
Ponto Crítico	Tamanho e estabilidade do produto
Medidas Utilizadas	Componentes
Decisões	Avaliar o número de componentes existentes dentro de um módulo, ou projeto.
Nome	Complexidade ciclomatica - tcta
Ponto Crítico	Qualidade do produto
Medidas Utilizadas	Complexidade ciclomatica - tcta
Decisões	Avaliar a evolução da complexidade nos componentes do projeto.

Quadro 18 - Parte 7 – Plano de mensuração – Classificação dos Indicadores
Fonte : PSM INSIGHT – INFRAERO

Parte 8 – Relatórios

São gerados dois tipos de relatórios de métricas:

 Gerencial - Relatório gerado mensalmente com as seguintes informações:

- Progresso do projeto
- Gastos
- Problemas críticos identificados

 Técnico - Relatório gerado quinzenalmente com as seguintes informações:

- Progresso do projeto
- Evolução do cronograma
- Problemas identificados
- Soluções implementadas

4.5. Resultados obtidos – Indicadores

Foram aplicadas as métricas descritas no plano de mensuração citada no projeto Internet Banking e os resultados dos indicadores foram:

- Horas trabalhadas equipe

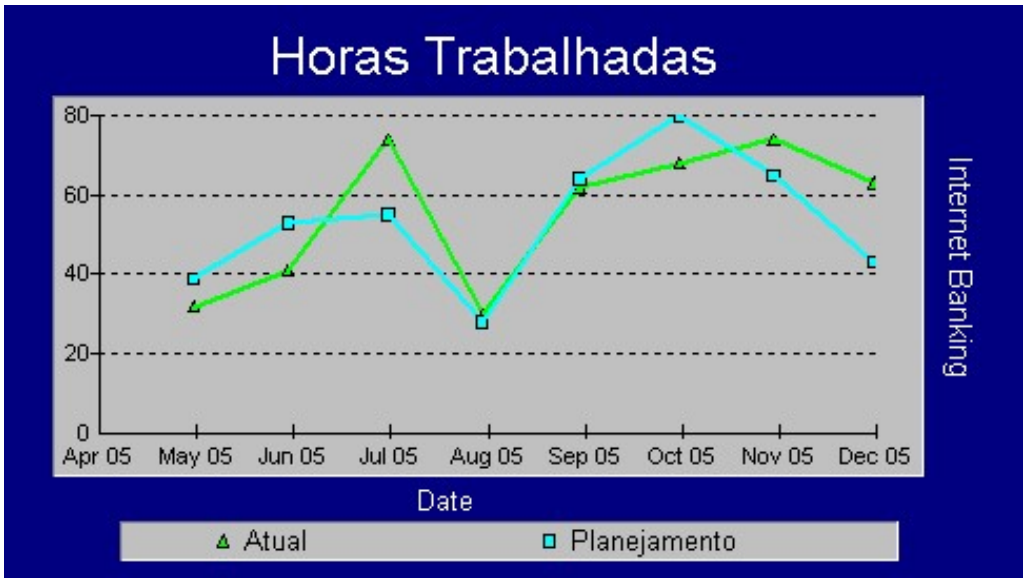


Figura 39 - Horas Trabalhadas
Fonte: PSM INSIGHT - INFRAERO

Na Figura 39 observa-se que nos meses de julho e novembro ocorreu um aumento das horas trabalhas devido a serem meses de entrega parcial e final do projeto.

Com um indicador como este o gerente de projeto poderia ter efetuado alterações no planejamento ou estruturado a equipe para que o projeto não fosse comprometido.

Este projeto foi entregue, porém, com uma grande carga de trabalho em seus limites de tempo; este tipo de atividade é arriscada, pois qualquer problema atrasa o projeto.

- Horas trabalhadas – Por Atividade / Por Mês

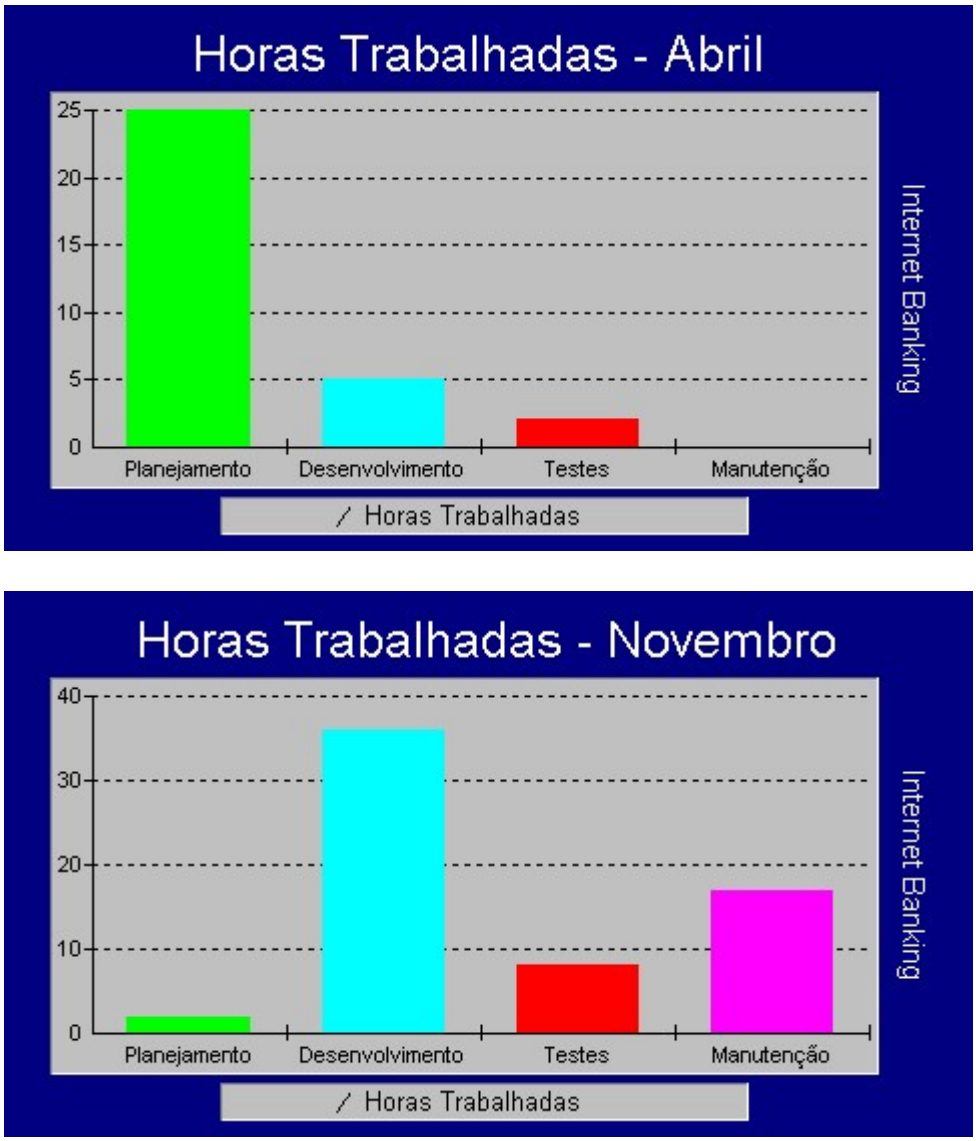


Figura 40 - Horas Trabalhadas / mês
Fonte: PSM INSIGHT - INFRAERO

Este indicador avalia as horas trabalhadas por atividade no mês.

Na Figura 40, observa-se que nos dois indicadores acima (abril – primeiro mês e novembro – último mês), o projeto seguiu um ciclo de desenvolvimento, sendo que no primeiro mês o planejamento foi a atividade mais trabalhada e no último mês o desenvolvimento.

Com um indicador como este o gerente de projeto pode avaliar como o esforço das horas trabalhadas está sendo dividido por atividades no mês.

Este resultado é muito importante para o controle do projeto, pois além de monitorar os esforços das atividades é possível monitorar atividades como manutenção que, se estiverem com valores altos, indicam problemas com o desenvolvimento.

- Horas trabalhadas – Por Atividade específica

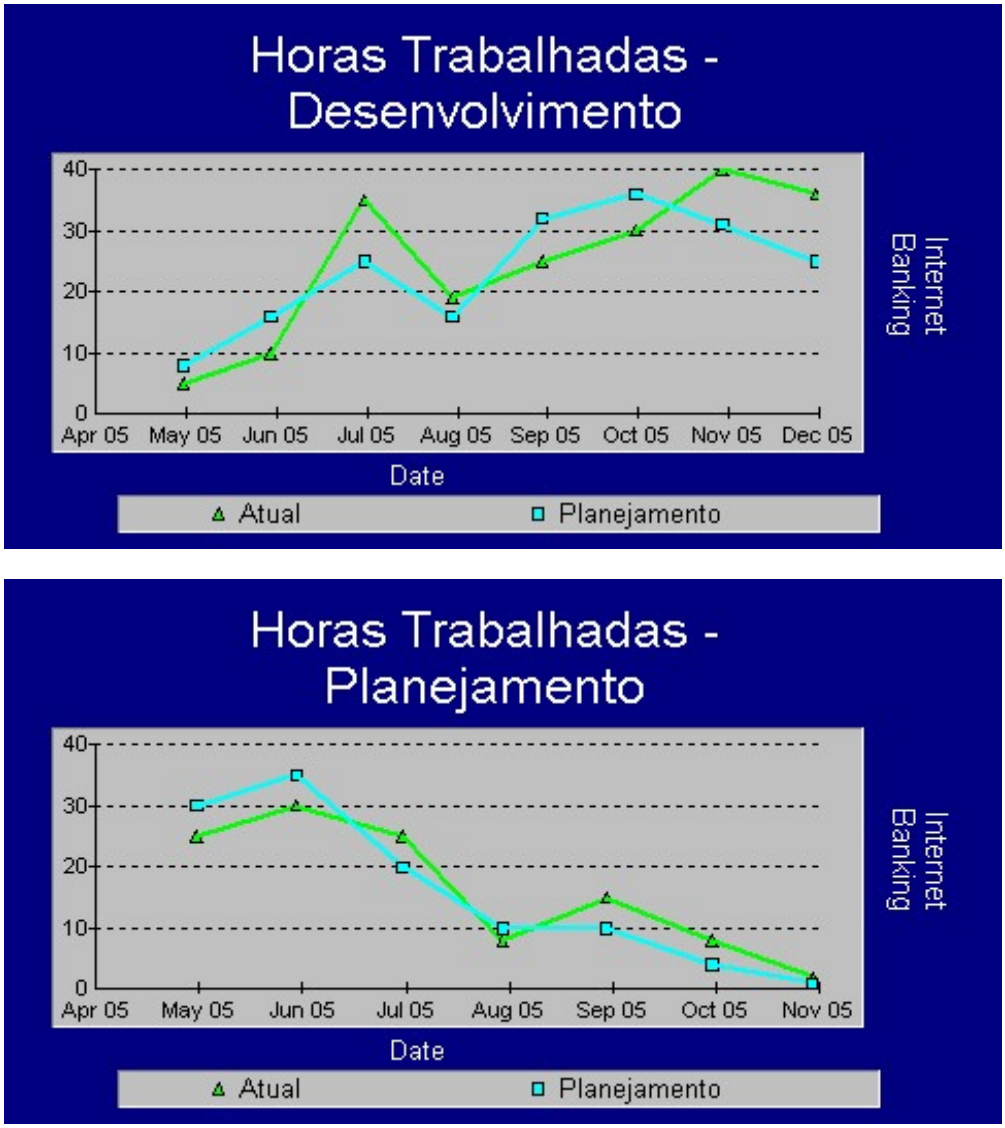


Figura 41 - Horas Trabalhadas / atividade
Fonte: PSM INSIGHT - INFRAERO

Este indicador mostra a evolução de uma atividade especifica durante todo o projeto comparando os valores realizados com os planejados. Na figura 41, observa-se que com os dois indicadores é possível determinar que ocorreu um aumento no final do projeto para as duas atividades. No caso do desenvolvimento seu aumento no final do projeto não chama atenção, pois nos meses anteriores ele não foi cumprido.

Já a atividade de planejamento aparece com valores acima do planejado durante os últimos três meses do projeto. Este resultado pode ser gerado por dois motivos:

- Problemas com o projeto, necessitando re-planejamento para a solução: Uma maneira de se comprovar esta possibilidade é verificando os índices de manutenção do sistema; se eles estiverem altos é possível que o planejamento tenha que passar por modificações.

- Problemas com a estimativa : O planejamento foi estimado de maneira errada.

- Horas trabalhadas – Por cargos

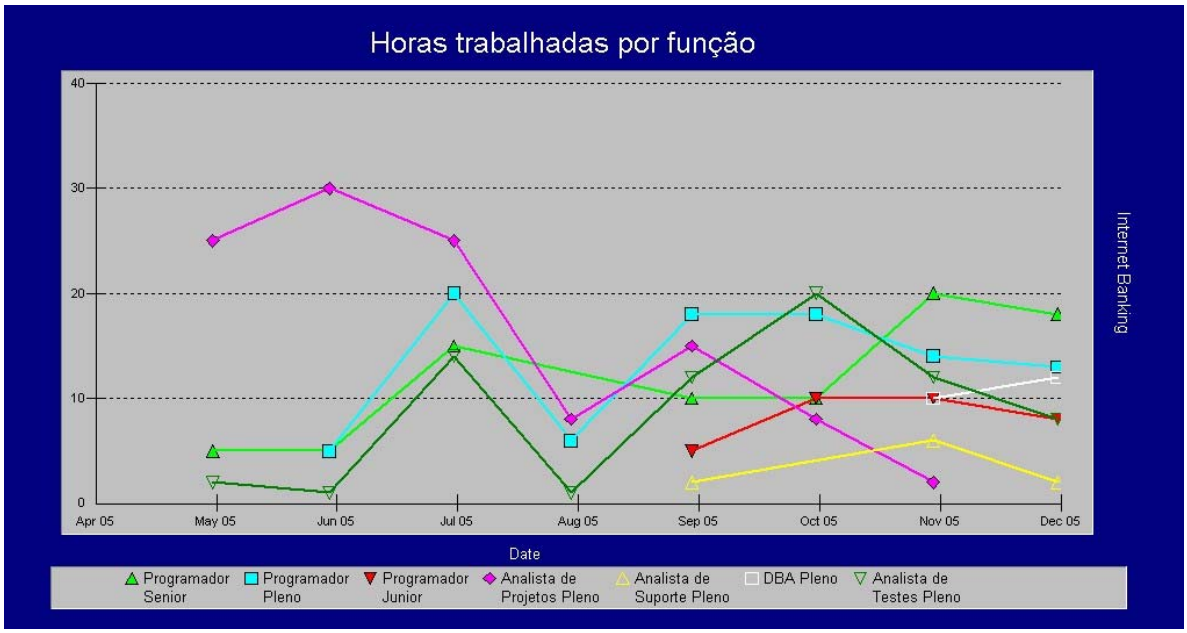
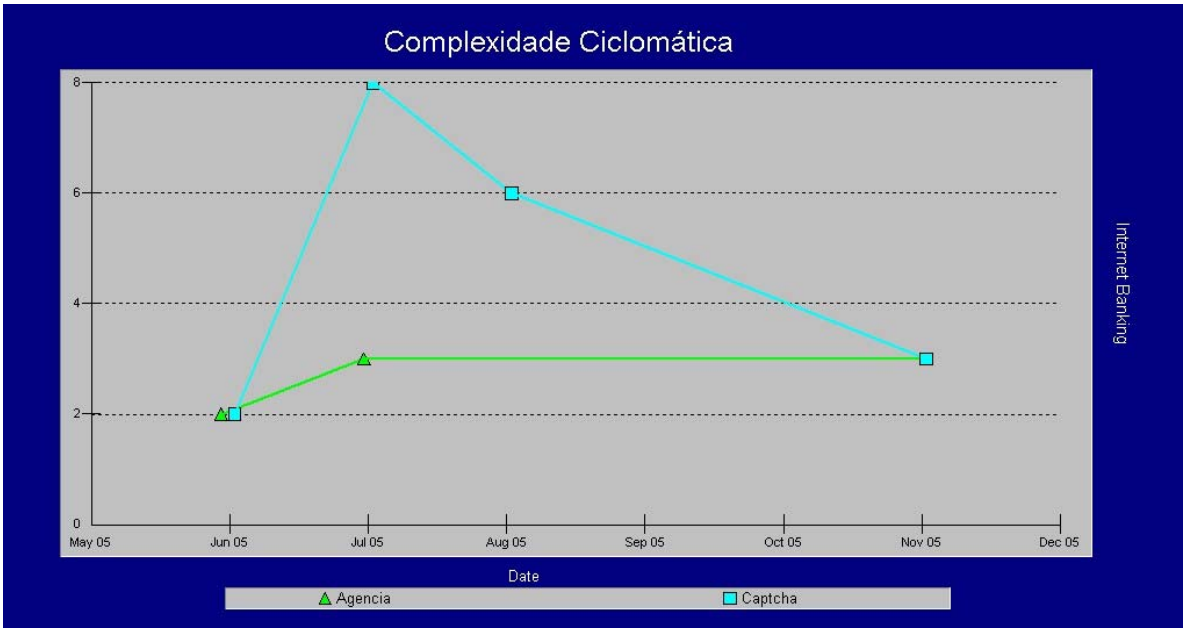


Figura 42 - Horas Trabalhadas / Cargo
Fonte: PSM INSIGHT - INFRAERO

Este indicador mostra as horas trabalhadas por cada um dos cargos durante o projeto. Com ele é possível determinar como está sendo o desempenho de cada função e avaliar quais funções estão sobrecarregadas.

- Indicador de Complexidade / Indicador de componentes.



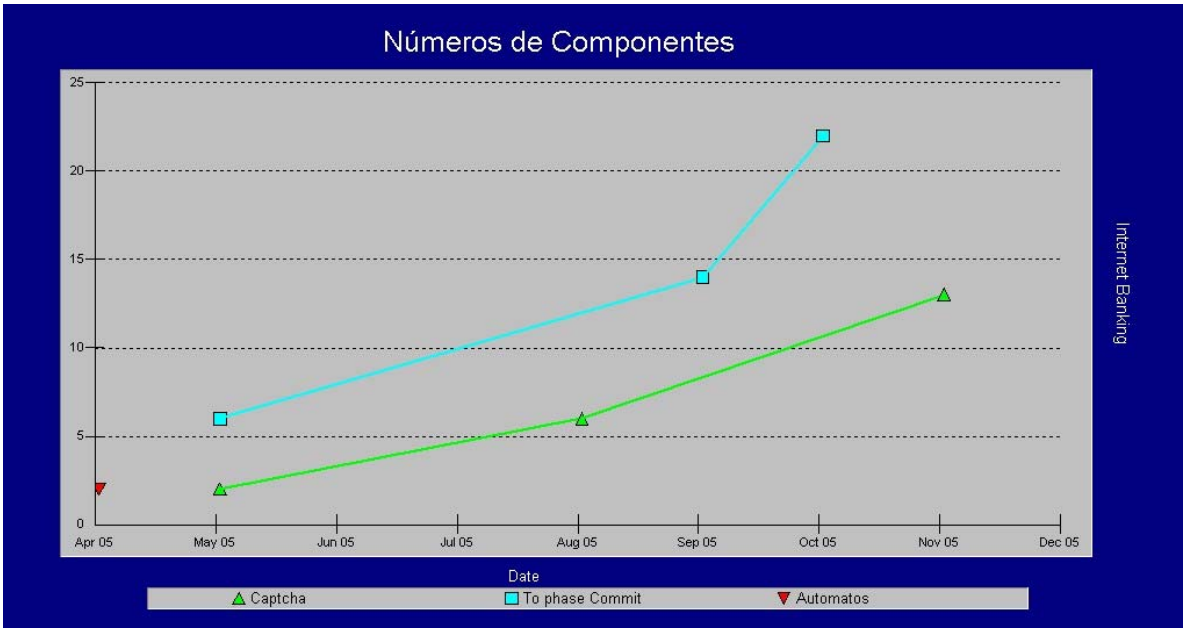


Figura 43 - Complexidade / Componentes
Fonte : PSM INSIGHT - INFRAERO

Estes indicadores mostram a evolução do desenvolvimento de um componente em número de classes e em complexidade de cada classe. Na Figura 43 observa-se que quando um componente atinge um valor de complexidade alto é necessário que ele seja modularizado, isto pode ser visto nos indicadores pelo componente *Captcha*; sua complexidade foi reduzida, porém, seu número de classes aumentou.

Estes indicadores são um monitoramento para o gerente agir quando um componente está se tornando mais complexo do que deve ser. Quanto mais simples os módulos forem mais manutenibilidade o software terá.

- Registro em banco de dados

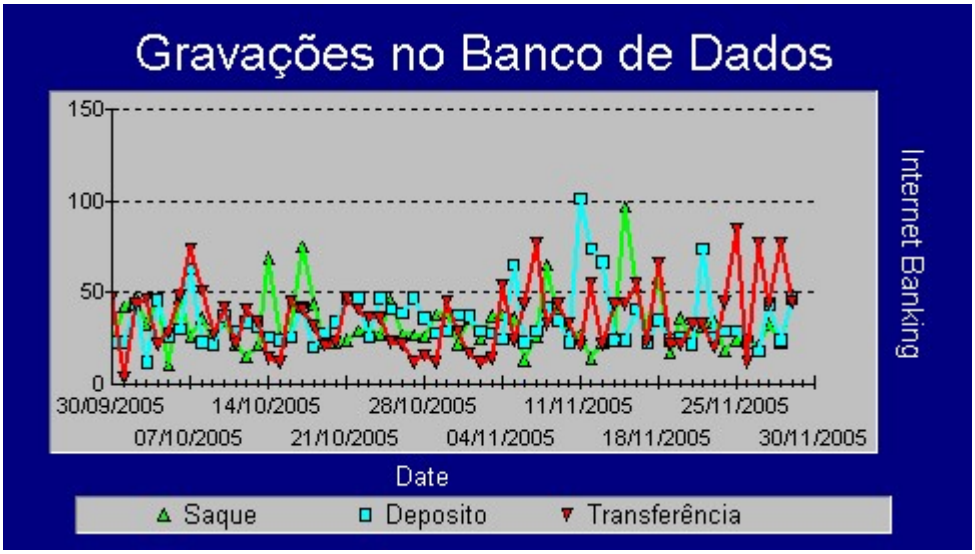


Figura 44 - Gravações no banco de dados
Fonte : PSM INSIGHT - INFRAERO.

Este indicador fornece ao gerente um monitoramento da carga utilizada pelo banco de dados. Para sistemas que fazem muitas operações com o banco, no caso o Internet Banking, é importante ter o controle para tomar ações no momento certo, por exemplo: a ampliação de espaço físico do banco, ou as melhoras na performance. A figura 44 demonstra a análise gráfica dessa indicação.

Como finalização deste caso foi possível observar que o modelo PSM INSIGHT está em franca expansão e tem sido considerado de extrema importância ao permitir a definição de estratégias de medição nos projetos de software.

CONCLUSÃO

Como pôde ser observado no estudo de caso, com base no critério de atingimento do escore mínimo para identificação do nível de maturidade, podemos concluir que o PSM INSIGHT pôde auxiliar a INFRAERO-RJ no processo de mensuração do desenvolvimento de um software, pelos critérios do CMMI. Ainda que indicadores fossem apresentados em níveis inferiores de maturidade, a instituição conseguiu observar com maior clareza dados como horas trabalhadas, problemas encontrados e componentes desenvolvidos, que caracterizam a não-satisfação das metas das áreas de processo neste nível de maturidade (CMMI nível 2).

Com o PSM melhorou-se muito: a qualidade no processo de desenvolvimento do software, os prazos bem acompanhados e permitindo a sinalização de uma posição crítica; a maior precisão, principalmente quando da utilização de períodos de contingência, os custos acompanhados e assistidos diretamente pelos profissionais envolvidos no processo em execução, através de gráficos divulgados a todas estações, ou seja, sim melhorou e muito a qualidade de desenvolvimento do software com a utilização do PSM.

Pelos resultados obtidos pode-se concluir, com 95% de certeza, que, segundo a visão dos executivos da organização, seu processo de desenvolvimento de software, especificamente o (IBI) Internet Banking Infraero-RJ, pode ser classificado como caminhando ao nível 2 de maturidade do CMMI, sendo, portanto, planejado, executado, medido e controlado; as práticas existentes são mantidas, mesmo nos momentos de crise, podendo repetir a experiência para novos projetos de software.

No entanto, a hipótese da pesquisa foi considerada parcialmente plausível, pois, os respondentes indicaram que o processo existente não pôde ser caracterizado como tendo os objetivos do departamento de desenvolvimento de software-web da organização baseados neste, nem operar dentro dos limites estabelecidos ou ser inovado na busca de oportunidades, características essas que garantiriam a aceitação dos usuários, desenvolvedores e demais *stakeholders*

A partir dos resultados encontrados conclui-se que a INFRAERO-RJ mostrou utilizar um método estruturado para desenvolvimento de seus softwares-web, o qual, no entanto, tem campo suficiente – no entendimento dos próprios gestores – de tornar-se mais completo, abrangente e potente para alavancar resultados de qualidade de software e financeiros da própria organização. Com uma estratégia previamente determinada pelo governo federal no desenvolvimento de software, requisitos de qualidade, dentre eles, custo, tempo, requisitos funcionais e não funcionais, foram satisfatoriamente atendidos, ou seja, com o PSM

INSIGHT, permitiu-se um melhor alinhamento do planejado ao realizado, resultando em um monitoramento e gerenciamento imperativo dentro de seu campo de análise. Para isso, observou-se que recomendamos a ação sobre os aspectos de integração e colaboração entre os envolvidos no desenvolvimento dos novos produtos (no segmento de softwares para web) e no envolvimento do usuário na verificação de componentes, processos e produto final, indicando serem pontos para melhoria no processo nos resultados da presente pesquisa.

A eficácia das ações a serem implementadas pode ser verificada em estudo futuro e, como refinamento do método e técnicas utilizados neste estudo, pelo crivo de avaliação lógica paraconsistente e/ou lógica fuzzy, de forma a dar maior respaldo e robustez às análises.

REFERÊNCIAS BIBLIOGRÁFICAS

AGUIAR, Maurício. PSM: *O CMM da mensuração de software?* TI métricas. A PSM Transition Organization. Disponível em http://www.metricas.com.br. Acesso em 15 set 2007.

ASR CONSULTORIA E ASSESSORIA EM QUALIDADE. Spin-Bh. *CMM – CMMI Principais conceitos, diferenças e correlações*. VOLPE, R.L.D; JOMORI, S.M.; ZABEU, Ana C.P. São Paulo, 2002.

CARDOSO, Caíque. *UML na prática: do problema ao sistema*. Rio de Janeiro: Ciência Moderna, 2003.

CLELAND, David I. *Gerência de projetos*. Rio de Janeiro: Reichmann & Affonso, 2002.

CMM. *Capability Maturity Model*. Desenvolvido pela SEI – *Software Engineering Institute*. Disponível em http://www.sei.cmu.edu. Acesso em 04 jan 2007.

_____. *Modelo de Maturidade de Capabilidade de Software*. Tradução de José Marcos Gonçalves e André Villas Boas Versão 1.2 datada de 11 out 2001.

CMMi. *Models and Modules*. Disponível em http://www.sei.cmu.edu/cmmi/models Acesso em 04 jan 2007.

CMU/SEI-TR-24-,1993. Disponível em http://www.sei.cmu.edu/pub/documents/95. reports/ps/tr012.95.ps Acesso em 04 abr 2007.

CMU/SEI-TR-010, 2002. Disponível em http://www.sei.cmu.edu/pub/documents/95. reports/ps/tr012.95.ps Acesso em 04 abr 2007.

CONFERÊNCIA LATINO-AMERICANA DE INFORMÁTICA - CLEI2004. *Modelo de Referência para Melhoria de Processo De Software: uma abordagem brasileira*. Arequipa, Peru. V1. 16p, 2004.

DoD**.** Report of the Defense Science Board Task Force on Military Sof tware, *Office of the Under Secretary of Defense for Acquisition*,Washington,D.C. September 1987.

GALARRABA, O. *Capabiblity Maturity Model: Um modelo para Melhoria do Processo de Produção de Software*, 2002. Disponível em http://www.pmirs.org/PMI21_Documentos.htm Acesso em 05 ago 2004.

GARCIA JR., P.R.; NUNES, D.J. *APSEE-Metrics: um modelo para mensuração em processos de software*. Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS) - Centro Universitário Ritter dos Reis (UNIRITTER), 2007.

GUIDE LINE CMMI_V1.2 – Agosto 2006. Disponível em http://www.guideline.com.br. Acesso em 03 ago 2007.

GUSTAFSON, David A. Métricas Orientadas a objetos. In: GUSTAFSON, David A. *Engenharia de Software*. São Paulo: Bookman, 2002, p.170-184.

HELDMAN, Kim. *Gerência de Projetos: guia para o exame oficial PMI*. Rio de Janeiro: Elsevier, 2003.

HUMPHREY, W.S. *Characterizing the software process*; A Maturity Framework, Software Engineering Institute, CMU/SEI-87-TR-11, ADA182895, June 1987.

_____. *A Comparison of U.S. and Japanese Software Process Maturity*. Proceedings of the 13 th International Conference on Software Engineering, Austin, TX,13-17 May 1991, p.38-49.

IBM SOFTWARE GROUP. *Student Manual*. REQ480 Mastering Requirements Management with Use Cases, Rational University, 2003.

IEEE - INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. An Evaluation of the MOOD Set Of Object-Oriented Software Metrics. Disponível em http://www.sce. carleton.ca/faculty/ajila/4106-5006/MOOD.pdf. Acesso em 21 mar 2006.

ISO/IEC-15504 (SPICE). Define um modelo bi-dimensional que tem por objetivo a realização de avaliações de processos de software com o foco da melhoria dos processos. Publicada em outubro de 2003.

JOSKO, J.M.B.; CÔRTES, M.L. P-CMM e outros modelos na Gestão de Pessoas. Instituto de Computação – Universidade Estadual de Campinas (UNICAMP) *VII Simpósio Internacional de Melhoria de Processos de Software*. São Paulo, SP – Brasil 21-23 nov 2005

KERZNER, Harold. *Strategic Planning for Project Management using a Project Maturity Model*. New York: John Wiley & Sons, 2001.

_____. *Gestão de projetos*: *As melhores práticas*. Porto Alegre: Bookman, 2002.

KOSCIANSKI, André. *Qualidade de Software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software.* São Paulo: Novatec, 2006.

McCABE, T. J. A complexity measure. In *ICSE'76: Proceedings of the 2nd international conference on Software engineering,* page 407, Los Alamitos, CA, USA, 1976.

McGARRY, Cardm Jones; LAYMAN, Clark; DEAN, Hall. *Pratical Software Measurement – Objective Information for Decision makers* – Addison-Wesley, 2002.

MECENAS, Ivan; OLIVEIRA, Viviane de. *Qualidade em Software: Uma metodologia para homologação de sistemas*. Rio de Janeiro: AltaBooks, 2005.

MICHAELIS. *Dicionário Prático: Língua Portuguesa*. São Paulo: Melhoramentos, 2002.

NAVAS, J.C.; ALVES, R.F. O MC2Q-SW – Um Método para a Melhoria Contínua da Qualidade de Software apoiado em CMM e SPICE - e sua Aplicação Experimental como base a Programas de Melhoria da Qualidade de Software.*VI Simpósio Internacional de Melhoria de Processos de Software.* São Paulo, SP – Brasil 24-26 nov 2004.

OGC - OFFICE OF GOVERNMENT COMMERCE (UK). What is PRINCE2?. Disponível em http://www.prince2.com/WHATISP2.HTML. Acesso em 14 maio 2006.

OPM3 – PROJECT MANAGEMENT INSTITUTE. Disponível em http://www.pmi.org.hk/ OPM3_ 050607_HKCS.pdf. Acesso em 09 abr 2006.

PAULK**,** M.C., CURTIS, B.; CHRISSIS, M.B. et al. *Capability Maturity Model for Software*, Software Engineering Institute, CMU/SEI-91-TR- 24,ADA240603,Agust 1991.

PETERS, James F.; PEDRYCZ, Witold. Medidas de software. In: PETERS, James F.; PEDRYCZ, Witold. *Engenharia de Software – Teoria e Prática.* São Paulo: Campus, 2001, p.429-498.

PMM - PROJECT MANAGEMENT MATURITY: In Industry Benchmark, USA. Project Management Journal. Project Management Institute 2003 34, pg.4-11.

PMMM. Disponível em http://www.e-programme.com/pmmm.htm. Acesso em 29 mar 2007.

PSM. PRATICAL SOFTWARE AND SYSTEMS MEASUREMENT. What is Model? Disponível em: http://www.pmsolutions.com/maturitymodel/whatismodel.htm Acesso em 09 abr 2006.

ROCHA, A.R.C.; MALDONADO, J.C.; WEBER, K.C. *Qualidade de Software: teoria e prática.* São Paulo:. Prentice Hall, 2001.

RODRIGUES, André Figueiredo. *Como elaborar referência bibliográfica*. 5 ed. Revisada e ampliada – São Paulo: Humanitas, 2004.

SALES, Rubens. *Curso de CMMI.* São Bernardo do Campo, 2005. 138 f. Trabalho de pesquisa. Setembro de 2005.

SCHMITZ, Eber Assis. *Modelo qualitativo de análise de risco: para projetos de tecnologia da informação*. Rio de Janeiro: Brasport, 2006.

SEI - SOFTWARE ENGINEERING INSTITUTE. *Software Acquisition capability maturity Model* (SA-CMM) Version 1.03.Disponível em: http://www.sei.cmu.edu/pub/ documents/02.reports/pdf/02tr010.pdf> . Acesso em 02 abr 2006.

_____. *CMM based appraisal for internal process improvement* (CBA IPI) - Team Training Material. U.S.A: Addison Wesley, 1996.

_____. *Process maturity profile of the software community* - year end update. U.S.A: Addison Wesley, 1997.

SIEGEL, Sidney. Estatística não-paramétrica para as ciências do comportamento. São Paulo: McGraw-Hill do Brasil, 1981.

SIEGEL, J. A. L. et al. National Software Capacity: Near –Term Study, Software Engineering Institute, CMU/SEI-90-TR-12-ADA226694,May 1990.

STEPHEN, Nelson L. *Project 2002: gerenciamento eficiente de projetos em oito etapas*. Rio de Janeiro: Ciência Moderna, 2003.

US ARMY. Pratical Software and Systems Measurement (PSM) – PSM Guide 4.0b1. Disponível em http://www.psmsc.com. Acesso em 02 abr 2006.

UNIVERSIDADE FEDERAL DE PERNANBUCO. Visão Geral do que é o CMM.Qualidade de Software, CIn – UFPE. Os Cinco Níveis de Maturidade. Disponível em http://cin.ufpe.br /~mjmcj/cmm/ visao_geral.html. Acesso em 02 abr 2006.

VARGAS, Ricardo Viana. *Manual prático do plano do projeto*. 2ª. ed. Atual. Rio de Janeiro: Brasport, 2005.

VIVACQUA, Flavio; XAVIER, Luiz Fernando da Silva. *Metodologia de gerenciamento de projetos*. Rio de Janeiro: Brasport, 2005.

WEBER**,** C.V.; PAULK, M.C.; WISE, C.J.; WITHEY, J.V. *Key Practices of the Capabilities Maturity Model, Software Engineering Institute*, CMU/SEI-91-TR-25,ADA240604,AUGUST 1991.

ZANATTA, A.L.; VILAIN, P. Uma análise do método ágil Scrum conforme abordagem nas áreas de processo Gerenciamento e Desenvolvimento de Requisitos do CMMI. Universidade Federal de Santa Catarina. *Anais do WER05 - Workshop em Engenharia de Requisitos*, Porto, Portugal, Junho 13-14, 2005, pp 209-220.

ANEXOS

Questionário

Visão Geral das Métricas na Organização utilizando o PSM INSIGHT

1. Identificação

Nome da Organização: _____
Área de Atuação: _____
Localização: _____
Responsável pelo Documento: _____
Produzido em: ___ / ___ / _____

2. Descrição Inicial da Organização

<Apresentar uma introdução ao documento, descrevendo rapidamente características gerais da organização, tais como: número de empregados, aeroportos, volume de negócios, tempo de existência>

3. Objetivos

<Descreve os principais objetivos da avaliação, indicando os fatores que foram observados durante o período de avaliação da organização e para que este artefato está sendo produzido>

4. Situação Atual

<Documenta a situação do processo de desenvolvimento e o resultado da avaliação da utilização de métricas na organização, focalizando aspectos de progresso na utilização do PSM INSIGHT>

4.1. Processo de Desenvolvimento

4.1.1. Características do Processo

<Apresenta as características do processo de desenvolvimento atualmente utilizado na organização: paradigma de desenvolvimento, pessoas envolvidas, papéis definidos, atividades relacionadas à produção de software, ciclo de vida, etc. Aqui deve ficar clara a maturidade da organização e dos membros da equipe, bem como a tecnologia utilizada durante o desenvolvimento de projetos de softwares.>

4.1.2. Principais Problemas no Processo

<Descreve os principais problemas observados na organização durante o desenvolvimento de sistemas: problemas de comunicação entre membros da equipe, falta de formalização do processo, deficiências técnicas. <Indicar, quando possível, planos de contingências>

4.2. Status da utilização de métricas

4.2.1. Métricas usadas na organização

<Indica a maturidade da organização quanto a utilização de métricas. Listar as métricas usadas, classificando-as, de acordo com o aspecto que ela captura (qualidade, progresso, tamanho) fazendo uso do PSM INSIGHT>

4.2.2. Problemas na utilização das métricas

<Lista de problemas relacionados à aplicação de métricas na organização: falta de estrutura para coleta de dados, métricas com eficiência não comprovada, mau uso dos resultados obtidos>