UNIVERSIDADE DO ESTADO DE SANTA CATARINA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VITOR MENDES

FERRAMENTA PARA DISTRIBUIÇÃO DE ESFORÇO PELAS FASES DE DESENVOLVIMENTO DE PROJETOS EM UM ESTUDO DE CASO

Edson Murakami

Professor Orientador

Edino Mariano Lopes Fernandes

Professor Co-orientador

Joinville, novembro de 2010

UNIVERSIDADE DO ESTADO DE SANTA CATARINA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VITOR MENDES

FERRAMENTA PARA DISTRIBUIÇÃO DE ESFORÇO PELAS FASES DE DESENVOLVIMENTO DE PROJETOS EM UM ESTUDO DE CASO

Trabalho de Conclusão de curso submetido à Universidade Estadual de Santa Catarina como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Edson Murakami

Professor Orientador

Edino Mariano Lopes Fernandes

Professor Co-orientador

Joinville, novembro de 2010

RESUMO

Atualmente existem dificuldades de se cumprir prazos e custos no desenvolvimento de software, onde uma das razões é a falta do uso de métodos de estimativa de projetos. O objetivo deste trabalho é desenvolver uma ferramenta para distribuição de esforço pelas fases do desenvolvimento de software da empresa PROSYST, criando um padrão para a estimativa de projetos utilizando como base a produtividade da empresa em projetos anteriores.

Palavras-chave: *Estimativa, Métricas, Esforço.*

ABSTRACT

Nowadays, there are difficulties to fulfill deadlines and costs in the development of software, one of the reasons is the lack of using methods for projects estimation. The objective of this work is to develop a tool for distribution of effort by stages of the development of software of the company PROSYST, using as a basis the company's productivity in projects issues.

Key words: *Estimation, Metric, Effort.*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A melhoria do processo de desenvolvimento de software tem demonstrado na prática ser uma abordagem viável, eficaz e eficiente para a necessária melhoria das organizações por meio da melhoria da capacidade de seus processos mais importantes.

A medição de desempenho é um fator essencial para a sobrevivência das empresas. Um bom gerente controla o desempenho de sua empresa por meio de indicadores de desempenho que, quando organizados, formam um sistema de medição de desempenho - SMD. Através deste sistema, a empresa pode verificar se está caminhando corretamente em direção aos objetivos definidos em sua estratégia. Uma empresa que não mede seu desempenho perde grandes oportunidades relacionadas a melhorias e identificação de oportunidades. Por fim, a empresa pode perder seu rumo, pois não sabe como está seu progresso em relação a aonde quer chegar.

Uma das questões fundamentais discutidas na área de engenharia de software é a produtividade dos programadores. Ao longo dos últimos anos, diversos estudos têm demonstrado uma grande disparidade na produtividade destes programadores diante do nível de experiência no ramo.

Em 1968, Sackman apud Peck (2006) apontou a razão entre o mais produtivo para o menos produtivo, na ordem de 28 para 1. Atualmente grande parte dos estudos sustenta esta razão na ordem de 10 para 1.

Estes dados impactam diretamente no setor de desenvolvimento dentro de uma organização, onde um programador com mais experiência pode produzir duas vezes mais em relação ao menos experiente, desta forma o planejamento do prazo e esforço para um dado projeto incorpora essas características para que de fato sejam cumpridos.

Atualmente a empresa PROSYST não utiliza uma metodologia para fazer as estimativas no desenvolvimento, o que de fato é utilizado é o método de estimativa com base na experiência do especialista para chegar ao esforço que deverá ser despendido.

Muitas vezes as estimativas condizem com o de fato realizado, normalmente em desenvolvimentos menores, que de fato já é de conhecimento prévio uma média de esforço necessária para se desenvolver o solicitado.

Considerando um projeto maior e mais robusto de funcionalidades, é fato que não é possível se fazer uma estimativa precisa para o desenvolvimento, onde frequentemente são consideradas menos horas do que será necessário. Como consequência, a empresa estará gastando seu próprio investimento na conclusão de um projeto o que, por sua vez, cria um impacto negativo na lucratividade da empresa.

1.1 OBJETIVOS

Este trabalho tem como objetivo prover uma ferramenta visando melhorar a precisão das estimativas de esforço no desenvolvimento de software da empresa PROSYST, utilizando como base a produtividade da mesma na razão do tamanho do projeto pelo esforço (em horas) utilizado.

1.2 METODOLOGIA

Para o estudo e obtenção dos resultados de produtividade, será realizado um levantamento bibliográfico, para se obter uma base teórica sobre os conceitos de engenharia de software e qualidade de software. Outra parte do estudo será realizada em cima da pesquisa dos métodos de avaliação de desempenho (métricas), com maior embasamento no método pontos por função.

Tendo esta base de conhecimento, é possível fazer um planejamento do estudo de caso, bem como o escopo (delimitações), definindo assim um método para avaliação por pontos de função.

Com o planejamento do estudo de caso e método definido, iniciará a parte prática do trabalho, onde serão realizadas as medições dos projetos para a obtenção da produtividade da empresa com o fim de realizar uma nova estimativa de esforço para outro projeto. Considerando que é de conhecimento prévio as horas utilizadas para o desenvolvimento desse novo projeto, é possível analisar os resultados obtidos e validar que a utilização da metodologia apresentada é eficaz.

Por fim, será desenvolvida uma ferramenta de estimativa e distribuição de esforço para novos projetos.

Esta ferramenta terá como entrada o número total de pontos de função calculados no projeto, que aliada à produtividade obtida, retorna um número total de horas para o projeto, ou seja, o esforço necessário para o desenvolvimento de um novo projeto.

Por fim, a ferramenta fará a distribuição das estimativas de horas nas fases do projeto apresentando de forma clara todas as etapas a serem cumpridas para os envolvidos.

1.3 JUSTIFICATIVA

A contribuição do trabalho é justamente a melhoria do processo de desenvolvimento, especificamente na estimativa de esforço, custo e prazo, além de fornecer uma ferramenta para a distribuição de esforço que esta baseada no próprio modelo de processo de desenvolvimento da empresa, além de fazer a calibragem da produtividade e percentual de esforço para cada fase automaticamente a cada projeto.

1.4 ESTRUTURA DO TRABALHO

A organização dos capítulos segue a linha da conceituação, problemática e por fim a solução proposta.

O capítulo 1 apresenta uma breve introdução à situação e sequencia do texto. O capítulo 2 apresenta os principais conceitos envolvendo engenharia de software e qualidade de software, este por sua vez levando aos conceitos de gestão de métricas de software, que é apresentado no capítulo 3. Neste capítulo é descrito alguns métodos utilizados na medição de software, além de uma comparação entre estes métodos para a escolha do que mais se adequa para os objetivos deste trabalho.

O capítulo 4 e 5, seguindo a organização dos capítulos, concentra-se em descrever a problemática encontrada no processo de desenvolvimento da empresa. Ainda neste capítulo é feita uma apresentação da empresa e a descrição de alguns processos que servem de auxílio para as estimativas de

esforço, além da identificação das fases do processo da empresa, medição do tamanho funcional de um projeto e a descoberta da produtividade da empresa.

O capítulo 6 descreve a solução proposta neste trabalho, que consiste na criação de uma ferramenta para distribuição de esforço pelas fases de desenvolvimento de software. Além disso, é apresentado o método de medição de software na prática, onde este é utilizado para encontrar a produtividade da empresa para um dado projeto.

Finalmente, o capítulo 7 apresenta uma análise dos resultados obtidos no trabalho, com uma conclusão e proposta para melhoramentos e trabalhos futuros dentro do contexto.

2 ENGENHARIA DE SOFTWARE

O desenvolvimento de software é uma atividade de crescente importância na sociedade contemporânea. A utilização de computadores nas mais diversas áreas do conhecimento humano tem gerado uma crescente demanda por soluções computadorizadas.

Falbo (2005) afirma que para os iniciantes na Ciência de Computação, desenvolver software é, muitas vezes, confundido com programação. Essa confusão inicial pode ser atribuída, parcialmente, pela forma como as pessoas são introduzidas nesta área de conhecimento, começando por desenvolver habilidades de raciocínio lógico, através de programação e estruturas de dados.

Esta estratégia não está errada, afinal se começa resolvendo pequenos problemas que gradativamente vão aumentando de complexidade, requerendo maior conhecimento e habilidade.

Entretanto, chega-se a um ponto em que, dado o tamanho ou a complexidade do problema que se pretende resolver, essa abordagem individual, centrada na programação não é mais indicada. De fato, ela só é aplicável para resolver pequenos problemas, tais como calcular médias, ordenar conjuntos de dados etc, envolvendo basicamente o projeto de um único algoritmo. Contudo, é insuficiente para problemas grandes e complexos, tais como aqueles tratados na automação bancária, na informatização de portos ou na gestão empresarial. Em tais situações, uma abordagem de engenharia é necessária (BOAS, 2003).

Observando outras áreas, tal como a Engenharia Civil, é possível verificar que situações análogas ocorrem. Por exemplo, para se construir uma casa de cachorro, não é necessário elaborar um projeto de engenharia civil, com plantas baixa, hidráulica e elétrica, ou mesmo cálculos estruturais. Um bom pedreiro é capaz de resolver o problema a contento.

Talvez não seja dada a melhor solução, mas o produto resultante pode atender aos requisitos pré-estabelecidos. Essa abordagem, contudo, não é viável para a construção de um edifício. Nesse caso, é necessário realizar um

estudo aprofundado, incluindo análises de solo, cálculos estruturais etc, seguido de um planejamento da execução da obra e desenvolvimento de modelos (maquetes e plantas de diversas naturezas), até a realização da obra, que deve ocorrer por etapas, tais como fundação, alvenaria, acabamento etc. Ao longo da realização do trabalho, deve-se realizar um acompanhamento para verificar prazos, custos e a qualidade do que se está construindo.

Visando melhorar a qualidade dos produtos de software e aumentar a produtividade no processo de desenvolvimento, surgiu a Engenharia de Software. A Engenharia de Software trata de aspectos relacionados ao estabelecimento de processos, métodos, técnicas, ferramentas e ambientes de suporte ao desenvolvimento de software.

Para tal, uma arquitetura deve ser estabelecida. Para apoiar a resolução de problemas, procedimentos (métodos, técnicas, roteiros etc) devem ser utilizados, bem como ferramentas para parcialmente automatizar o trabalho.

Neste cenário, muitas vezes não é possível conduzir o desenvolvimento de software de maneira individual. Pessoas têm de trabalhar em equipes, o esforço tem de ser planejado, coordenado e acompanhado, bem como a qualidade do que se está produzindo tem de ser sistematicamente avaliada.

Para Vasconcelos (2004), a engenharia de software não engloba apenas o desenvolvimento de programas, mas também toda a documentação necessária para o desenvolvimento, instalação, uso e manutenção dos programas. O termo “ciclo de vida de software” compreende todas as etapas, desde a concepção inicial do software, até a sua implementação, implantação, uso e manutenção, de modo que, ao final de cada uma destas etapas, um ou mais documentos serão produzidos.

2.1 MODELOS DE CICO DE VIDA

A fim de facilitar o entendimento do processo de desenvolvimento de software, vários modelos de ciclo de vida têm sido propostos. Estes modelos são descrições abstratas do processo de desenvolvimento, tipicamente

mostrando as principais atividades e dados usados na produção e manutenção de software, bem como a ordem em que as atividades devem ser executadas.

Entre os principais modelos existentes estão o modelo cascata, o modelo de desenvolvimento evolucionário, o modelo de transformação formal, o modelo de desenvolvimento baseado em reuso e os modelos iterativos.

Neste trabalho será abordado o modelo de cascata, sendo escolhido pelo fato de ser o modelo utilizado no processo de desenvolvimento da empresa.

2.1.1 MODELO CASCATA

No modelo cascata, o desenvolvimento de um software se dá de forma sequencial, a partir da atividade de verificação da viabilidade do desenvolvimento. Para cada etapa cumprida, segue-se a etapa imediatamente posterior, daí a ideia de uma cascata (Figura 2-1).

Este modelo tornou-se incompatível com a realidade atual porque a busca de requisitos não deve ocorrer somente no momento inicial do projeto, pois na maioria das vezes é praticamente impossível obter-se a totalidade de requisitos de maneira antecipada e em uma única etapa de um ciclo. Faz-se importante possibilitar o retorno para esta etapa sempre que for necessário. Outra característica refere-se especialmente aos testes que somente ocorrerão ao final de todo processo, onde um erro sutil pode vir a exigir semanas de verificação para que se possa eliminá-lo, e com isso atrasar eventuais cronogramas (SHAW, 1990).

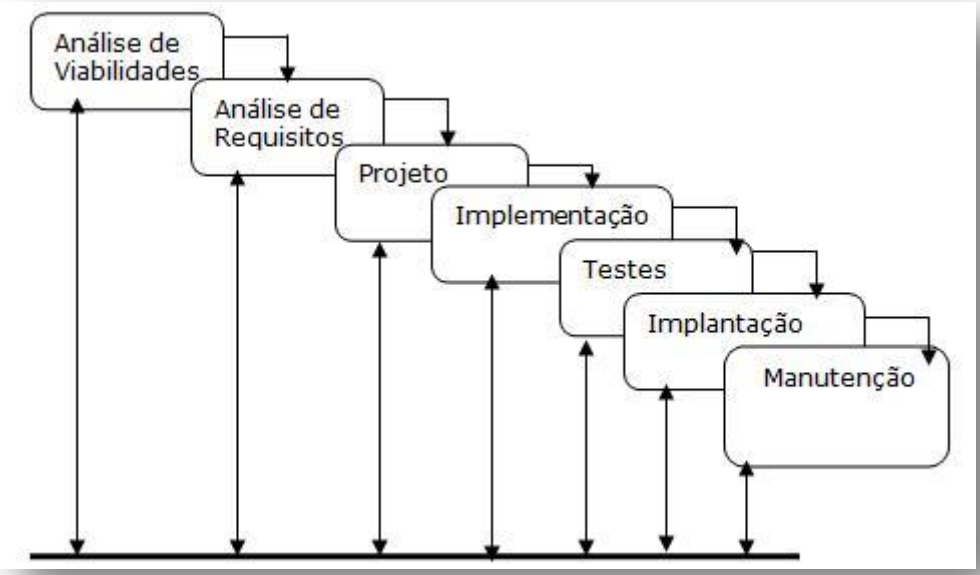


FIGURA 2-1 MODELO CASCATA (ADAPTADO DE VASCONCELOS, 2003).

Este modelo foi derivado de modelos existentes em outras engenharias e considera que o processo de desenvolvimento de software é composto por várias etapas que são executadas de forma sistemática e sequencial. Estas etapas são descritas a seguir:

- **Estudo de Viabilidade:** tem como objetivo delinear o escopo do problema a ser resolvido; identificar alternativas de solução, identificar seus custos e tempo para execução; identificar potenciais benefícios para o usuário.
- **Definição de Requisitos:** durante esta etapa os serviços, as metas e as restrições impostas ao sistema são identificados junto aos usuários do software, além de analisados de modo a remover inconsistências e ambiguidades.
- **Projeto do Sistema e do Software:** nesta etapa os requisitos identificados são mapeados em componentes de hardware e software, de modo que o sistema possa ser posteriormente implementado. A arquitetura geral do sistema também é estabelecida.

- **Implementação e Testes Unitários:** nesta etapa o projeto de software é implementado em unidades de programas, utilizando-se uma linguagem de programação. Nessa etapa, as unidades implementadas também são testadas para assegurar a conformidade em relação às suas especificações.
- **Integração e Teste dos Sistemas:** nesta etapa as unidades de programas são integradas e testadas como um sistema completo, para assegurar que todos os requisitos do software foram atendidos. Recomenda-se que os testes sejam feitos à medida as unidades individuais vão sendo integradas (testes de integração) e que, ao final da integração, o sistema completo seja testado novamente (teste de sistema). Por este motivo, muitas vezes a integração é considerada como parte integrante da implementação.
- **Operação e Manutenção:** nesta etapa o sistema é instalado e colocado em operação. Posteriormente, quando erros forem encontrados no sistema ou quando forem solicitadas mudanças nos requisitos, o sistema entra numa etapa de manutenção.

Na sua forma mais simples, o modelo cascata não apresenta iterações, como visto na Figura 2-1, porém o processo de desenvolvimento de software pode ter etapas que são desenvolvidas em paralelo e de forma interativa, pois durante uma determinada etapa, problemas existentes na etapa anterior podem ser descobertos, como por exemplo, novos requisitos podem ser descobertos durante a realização da etapa de projeto, o que implica uma iteração para a etapa especificação e análise de requisitos).

Segundo Kezner (1998), a principal desvantagem do modelo cascata é que boa parte do sistema não estará disponível até um ponto adiantado do cronograma do projeto e, geralmente, é difícil convencer o cliente de que é preciso paciência. Além disso, existe a dificuldade de acomodação das mudanças depois que o processo está em andamento. Portanto, esse modelo é mais apropriado quando os requisitos são bem entendidos.

Existem algumas vantagens associadas ao modelo, pois ele oferece uma maneira de tornar o processo mais visível, fixando pontos específicos para a escrita de relatórios e, didaticamente, é uma maneira mais fácil de introduzir os principais conceitos de engenharia de software.

Os modelos de ciclo de vida de software são descrições abstratas do processo de desenvolvimento, tipicamente mostrando a sequencia de execução das principais atividades envolvidas na produção e evolução de um sistema de software. Com esta sequencia de atividades é possível definir as fases de um projeto, onde na qual, será realizada a distribuição de esforço.

2.2 QUALIDADE E NORMAS DE QUALIDADE DE SOFTWARE

A excelência no processo de desenvolvimento de software está diretamente ligada à qualidade do mesmo. Portanto antes de focar diretamente no objeto de gestão de métricas da engenharia de software, é importante relevar alguns conceitos sobre qualidade de software e normas de qualidade e como são incorporados nas empresas.

Nesta seção, estarão sendo explorados os mais importantes conceitos sobre os temas: Qualidade e Normas de Qualidade de Software.

2.2.1 QUALIDADE

A Norma ISO 8402 (ISO8402, 1993) define Qualidade como a “totalidade de características de uma entidade que lhe confere a capacidade de satisfazer as necessidades explícitas e implícitas”. Necessidades explícitas são aquelas expressas na definição de requisitos propostos pelo produtor. Esses requisitos definem as condições em que o produto deve ser utilizado, seus objetivos, funções e o desempenho esperado.

As necessidades implícitas são aquelas que, embora não expressas nos documentos do produtor, são necessárias para o usuário. Estão englobados nesta classe tanto os requisitos que não precisam ser declarados por serem óbvios (ex: “a caneta escreve”) como aqueles requisitos que não são percebidos como necessários no momento em que o produto foi desenvolvido, mas que pela gravidade de suas consequências devem ser atendidos (ex:

mesmo em condições não previstas, de erro ou má operação, um sistema de administração hospitalar não pode provocar a morte de pacientes).

Kano et al (KANO, 1984) classificam os requisitos de acordo com duas dimensões, conforme ilustrado no diagrama da Figura 2-2 em que o "atendimento ao requisito" é colocado em um eixo e o "sentimento de satisfação" no outro. Segundo essa visão, os requisitos de qualidade são classificados como:

- Necessários, se o não atendimento gera insatisfação e o atendimento gera indiferença. Por exemplo, a não ocorrência de defeitos em um automóvel nos primeiros 6 meses de uso, quando é o mínimo que se espera de um automóvel novo.
- Normais, se há insatisfação pelo não atendimento e satisfação pelo atendimento. Por exemplo, a economia, nível de ruído e o desempenho do automóvel.
- Atrativos, se provocam um sentimento de satisfação quando atendidos, mas de indiferença se não for atendido. Por exemplo, ar condicionado, direção hidráulica, "air bag" em um automóvel.

Deve-se notar que a aplicação desta classificação é dinâmica, sofrendo variações conforme o amadurecimento do mercado. Assim, o ar condicionado e a direção hidráulica já estão se tornando requisitos normais no mercado brasileiro e já são necessários em países desenvolvidos. No mercado de software, tanto a velocidade de amadurecimento do público como a introdução de inovações tecnológicas são muito maiores.

A análise destes requisitos segundo esse critério é muito útil para qualquer empresa que trabalhe em um mercado competitivo.

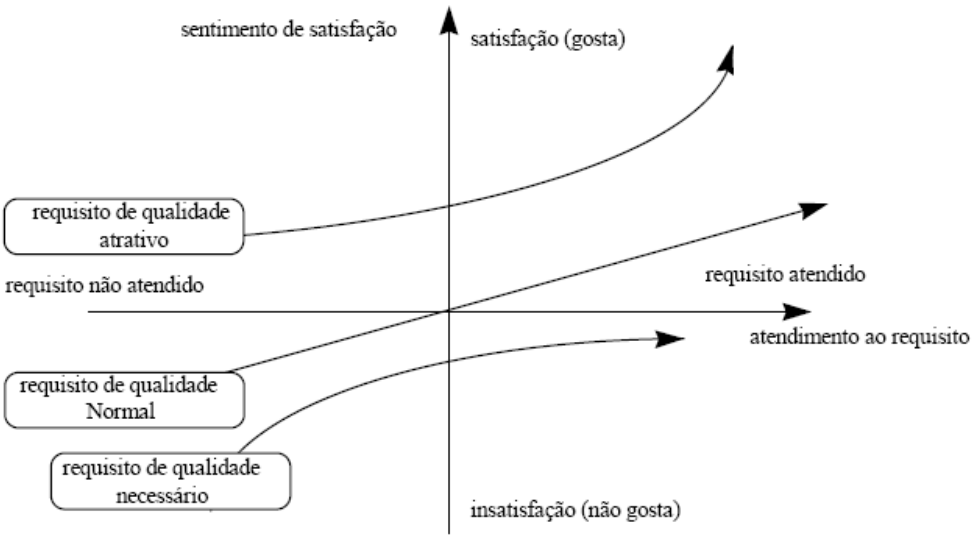


FIGURA 2-2 REQUISITOS DE QUALIDADE (WEBBER, 1999).

2.2.2 ENGENHARIA, QUALIDADE DE PROCESSO

A engenharia pode ser vista como uma confluência de práticas artesanais, comerciais e cientificas (SHAW, 1990). Em um primeiro momento, essas vertentes visam codificar o processo de geração de um produto. Em geral, apesar da preocupação com a qualidade estar subjacente ao desenvolvimento das técnicas de produção, a necessidade da avaliação e julgamento da qualidade do produto só é explicitado em fases posteriores, quando métodos para geração do produto já estão consolidados.

O controle de qualidade surge então, como uma necessidade. De inicio, são realizadas verificações esparsas e não sistemáticas, em seguida, adotam-se técnicas e critérios bem definidos, podendo, em alguns casos, chegar-se à verificação de 100% dos produtos para eliminação daqueles produzidos com defeito, impedindo que eles cheguem ao usuário. Certifica-se a qualidade do produto, a um custo elevadíssimo, seja pelo trabalho de verificação envolvido, seja pelo desperdício representado pela detecção e eliminação das peças defeituosas.

Como alternativa, procura-se melhorar o processo de produção, para se adquirir maior confiança na qualidade do produto final. Adota-se o Controle

Estatístico de Processo (CEP), para identificar variações no processo de forma a corrigir desvios.

No passo seguinte, adota-se a noção de um Sistema da Qualidade, envolvendo toda a empresa no esforço pela qualidade (a série de Normas ISO 9000 se insere neste passo). A seguir amplia-se ainda mais o alcance do gerenciamento da qualidade, com a necessidade de se fazer um projeto voltado para a qualidade e se considerar a satisfação de todos os agentes envolvidos: o cliente (produto ou serviço), o acionista (resultados financeiro), os colaboradores (emprego, crescimento profissional) e a comunidade (FEIGENBAUM, 1991).

Adota-se a visão de que a Qualidade se obtém principalmente durante o projeto e concepção do produto, tornando-o mais robusto, ou seja, menos sujeito a introdução de defeitos no processo de manufatura e menos passível de falhas conforme as condições ambientais e forma de uso. A Figura 2-3 mostra a evolução da proporção da aplicação de inspeção, atuação no processo e no projeto.

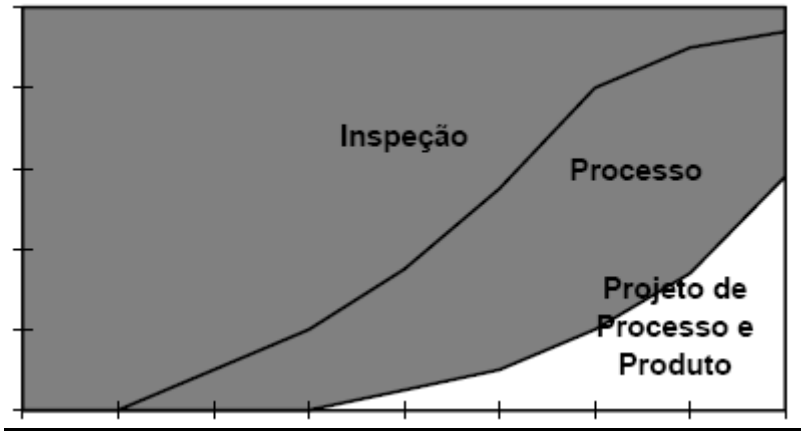


FIGURA 2-3 EVOLUÇÃO DA APLICAÇÃO DE MÉTODOS PARA A QUALIDADE COM O PASSAR DO TEMPO (BOAS, 2003).

A Figura 2-4 mostra a evolução dos conceitos de Qualidade ao longo dos anos.

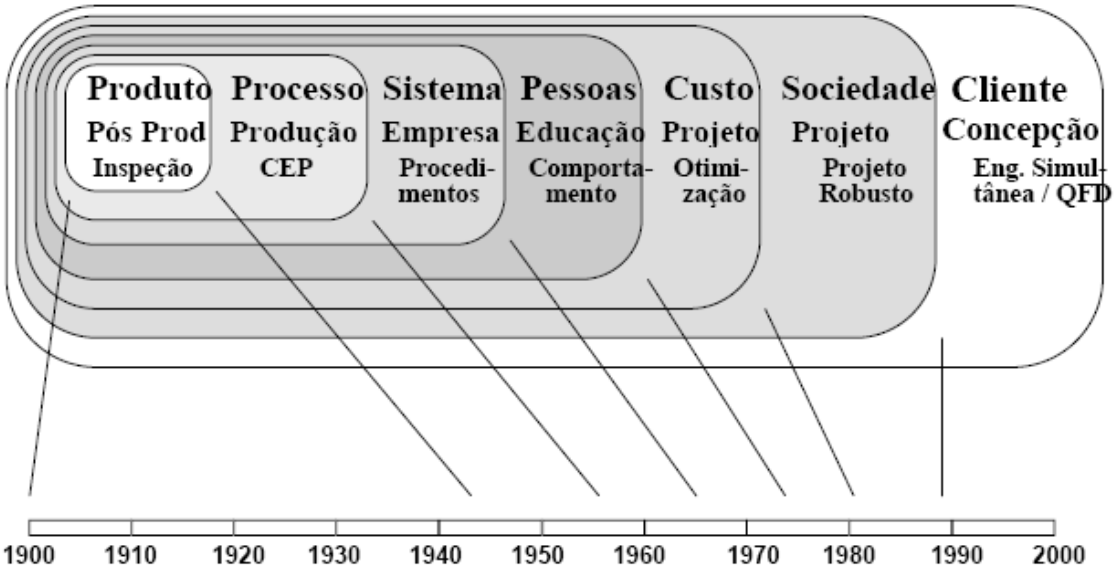


FIGURA 2-4 EVOLUÇÃO DOS CONCEITOS DE QUALIDADE (BOAS, 2003).

A Engenharia de Software tem como objetivo a melhoria da qualidade do seu produto, com propostas de modelos de desenvolvimento, métodos e técnicas para aplicação nas diversas fases de desenvolvimento do software. A avaliação da qualidade de software, nas duas visões (processo e produto), se insere nesse esforço.

Como na produção material, não cabe a dúvida quanto a se avaliar e julgar processo ou produto. As duas abordagens são necessárias e complementares. A visão de processo de software propicia uma estrutura para a harmonização das várias disciplinas da engenharia de software, englobando não apenas as atividades de desenvolvimento, mas todas as atividades necessárias para a sua produção, incluindo a avaliação.

2.2.3 QUALIDADE DESENVOLVIMENTO DE SOFTWARE

A qualidade de software é largamente determinada pela qualidade dos processos utilizados para o desenvolvimento. Deste modo, a melhoria da qualidade de software é obtida pela melhoria da qualidade dos processos. Esta visão orientou a elaboração de modelos de definição, avaliação e melhoria de processos de software (PRESSMAN, 2005).

A Figura 2-5 mostra a evolução das normas de qualidade ao longo dos anos em ordem cronológica. Os modelos mais significativos estão apresentados nos subitens a seguir.

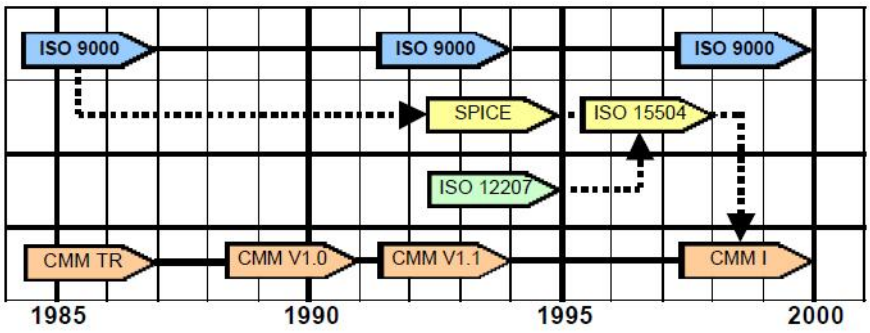


FIGURA 2-5 CRONOLOGIA DAS NORMAS DE QUALIDADE (SHEARD, 1997).

2.2.3.1 SPICE (Software Process Improvement and Capability Determination)

Em 1993 foi lançado o projeto SPICE, com o objetivo de gerar normas para avaliação de processos, visando à melhoria contínua do processo e a determinação da sua capacitação.

O modelo de referência do SPICE é na verdade um framework para avaliação de processos de software que harmoniza os diversos modelos nos quais ele se baseia, como: SW-CMM, Trillium, Software Technology Diagnostic (STD) e Bootstrap.

O objetivo é que cada um destes modelos, e outros que venham a ser criados possa ser definido como modelos compatíveis com este framework, possibilitando que os resultados de avaliações, segundo cada um destes modelos, possam ser comparados. A partir de 1998, é conhecido com o nome de norma ISO/IEC 15504.

O SPICE pode ser utilizado por organizações envolvidas em planejar, gerenciar, monitorar, controlar e melhorar a aquisição, fornecimento, desenvolvimento, operação, evolução e suporte de software.

Dentro da visão do SPICE (Figura 2-6), a avaliação de processos de software tem como propósito:

- Entender o estado dos processos de uma organização para a sua melhoria;
- Determinar a adequação dos processos de uma organização para um requisito particular ou uma classe de requisitos;

- Determinar a adequação dos processos de outra organização para um determinado contrato ou para uma classe de contratos.

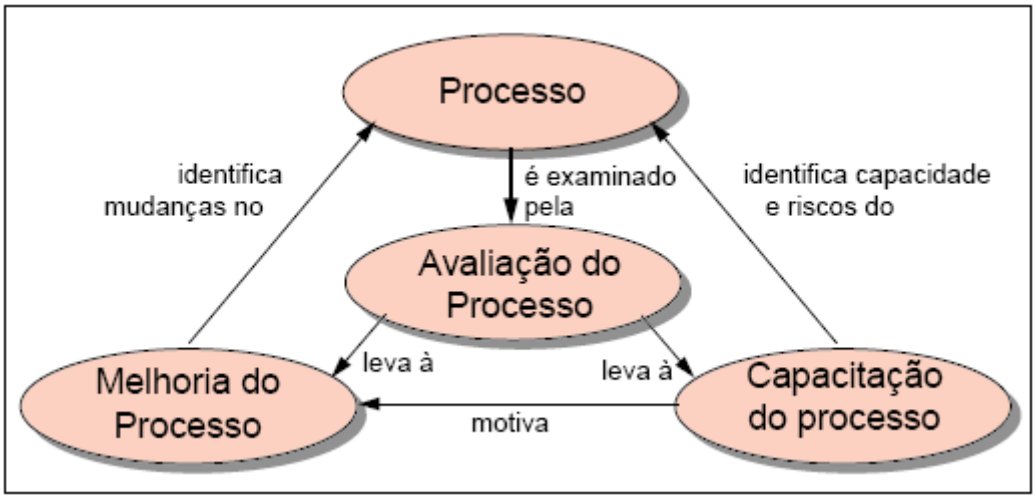


FIGURA 2-6 AVALIAÇÃO DE PROCESSO DE SOFTWARE - SPICE (SALVIANO, 2004).

Este projeto é interessante pelo seu direcionamento e flexibilidade. Está disponível para que as organizações o utilizem conforme suas necessidades e planos de negócios, medindo a capacitação de cada um de seus processos com o objetivo de promover melhorias contínuas nos mesmos.

Deste modo, obtém-se uma avaliação mais detalhada do estado da organização, permitindo a comparação de resultados de avaliações por outros modelos compatíveis.

2.2.3.2 CMMI

A sigla CMMI representa as iniciais de Capability Maturity Model Integration e nomeia tanto um projeto, quanto os modelos resultantes deste projeto.

O projeto CMMI, que pode ser traduzido como “Projeto de Integração dos Modelos de Maturidade da Capacidade”, é executado pelo Software Engineering Institute (SEI), em cooperação com a indústria e governo, para consolidar um framework para modelos, evoluir e integrar modelos

desenvolvidos pelo SEI (SW-CMM, SE-CMM e IPD-CMM), e gerar seus produto associados.

Esta integração e evolução tiveram como objetivo principal a redução do custo da implementação de melhoria de processo multidisciplinar baseada em modelos. Multidisciplinar porque além da engenharia de software, o CMMI cobre também a engenharia de sistemas, aquisição, e a cadeia de desenvolvimento de produto.

A redução de custo é obtida por meio da eliminação de inconsistências, redução de duplicações, melhoria da clareza e entendimento, utilização de terminologia comum, utilização de estilo consistente, estabelecimento de regras de construção uniforme entre outros (SALVIANO, 2004).

O CMMI possui duas representações: "contínua" ou "por estágios". Estas representações permitem a organização utilizar diferentes caminhos para a melhoria de acordo com seu interesse.

O modelo CMMI representa uma combinação da evolução do SW-CMM, que é o modelo que estabeleceu a área de melhoria de processo, com características da ISSO/IEC 15504.

As organizações de desenvolvimento de software podem utilizar um destes dois modelos, ou mesmo uma combinação dos dois, como uma referência para orientar ações para alteração dos processos utilizados para aquisição, fornecimento, desenvolvimento, manutenção e/ou suporte de sistemas de software, com o objetivo de estabelecer processos que satisfaçam de forma mais eficiente e eficaz os objetivos e necessidades de negócio da organização.

2.2.4 INDICADORES DE DESEMPENHO

A gestão de qualidade requer o estabelecimento de objetivos de médio, e de metas de curto prazo a serem cumpridas. Os objetivos e metas podem ser referir a produtos, serviços e processos.

Segundo Spinola (2004), os indicadores de desempenho são as informações que devem ser analisadas para saber se as metas e objetivos da qualidade estão sendo atingidos.

Com base nos indicadores, pode-se medir o progresso, aplicar ações corretivas quando necessário e planejar melhoria para o sistema de gestão da qualidade, aumentando o desempenho da organização. Sem os indicadores, as decisões tornam-se mais difíceis, o que dificulta planejar melhorias para o sistema.

Para Valle (2006), os indicadores de desempenho basicamente servem para:

- Definir objetivos e as metas da organização e, como conseqüência, de cada processo;
- Acompanhar o desempenho dos processos e, como conseqüência, de toda a organização;
- Identificar as áreas onde devem ser feitas ações corretivas, ou de melhoria;
- Eventualmente, redefinir objetivos e metas.

Ainda segundo Valle (2006), o objetivo desta avaliação é provocar uma resposta adequada dos indivíduos e grupos, isto é, que leve a melhorias nos processos. Este *feedback* é, portanto, indispensável para que o grupo saiba onde deve concentrar suas atenções e energias e, sobretudo, para que ele se motive para um aperfeiçoamento contínuo.

A abordagem deste trabalho será na geração de uma ferramenta para distribuição do esforço com base nos indicadores de produtividade da empresa no processo de desenvolvimento de software, esta ferramenta auxiliando no monitoramento, planejamento de novos projetos com base no próprio histórico da empresa.

Para obter-se o indicador de produtividade é necessário aliar o esforço despendido no projeto ao tamanho do mesmo. O esforço é dado por

indicadores de apontamentos da empresa, que serão apresentados no capítulo 4. Para a obtenção do tamanho funcional de um programa é necessário utilizar a metodologia de medição de software.

As métricas de processo e de projeto de software, segundo Pressman (1988), são medidas quantitativas que permitem aos engenheiros de software ter ideia da eficácia do processo de software.

Dados básicos de qualidade e produtividade são coletados e então analisados, comparados com medidas anteriores e avaliados para determinar se ocorrem melhorias de qualidade e produtividade.

Para Pressman (2005), as métricas também são utilizadas para detectar áreas de problema, de modo que soluções possam ser desenvolvidas, e que o processo de desenvolvimento de software possa ser melhorado.

CONSIDERAÇÕES PARCIAIS

A utilização dos conceitos da engenharia de software aumenta a probabilidade de produzir software de grande porte com qualidade, ou seja, software que satisfaça os requisitos do usuário, bem como as expectativas de tempo e do orçamento.

A avaliação de produtos de software tem sido uma das formas empregadas por organizações que produzem ou adquirem software para obtenção de maior qualidade nestes produtos, sejam eles produtos completos ou partes a serem integradas em um sistema computacional mais amplo.

A avaliação de desempenho também merece destaque para que a organização possa ter a visão de como melhorar o processo de desenvolvimento deste produto.

A utilização dos indicadores de desempenho agrega a esta avaliação, onde se cria a possibilidade de realizar comparativos e muitas vezes detectar qual parte do processo necessita de melhora.

No próximo capítulo serão abordados os principais conceitos das métricas de software, onde o objetivo é apresentar o índice de produtividade no desenvolvimento de um projeto com a utilização de um método quer será

apresentado. As métricas fazem parte do estudo de engenharia de software, porem a importância do mesmo para o trabalho resultou em um novo capítulo para destacar os conceitos.

3 MÉTRICAS

Uma das questões fundamentais discutidas na área de engenharia software é a produtividade dos programadores. Ao longo dos últimos anos, diversos estudos têm demonstrado uma grande disparidade na produtividade de programadores que possuem experiência no ramo.

Em 1968, Sackman apud Peck (2006) apontou a razão entre o mais produtivo para o menos produtivo, na ordem de 28 para 1. Atualmente grande parte dos estudos sustenta esta razão na ordem de 10 para 1.

Além da disparidade entre os níveis de produtividade, as pesquisas indicam que os programadores, considerados mais produtivos, também geram códigos de maior qualidade e eficiência.

Estes dados podem impactar diretamente no setor de desenvolvimento dentro de uma organização, onde um programador de alta produtividade, considerando a razão (10:1) aceita no mercado atualmente, pode ter uma produtividade no desenvolvimento de software duas vezes maior em relação a outro programador considerado “regular”, além disso, este mesmo programador provavelmente estará gerando um código de maior qualidade e eficiência, o que poupa o tempo gasto na execução de um projeto (PECK, 2006).

Parte do estudo deste trabalho está focada na medição da produtividade no desenvolvimento de software, porém não se faz menos importante levantar os aspectos que compõem as métricas de um projeto como um todo.

As próximas seções visam estabelecer uma relação das métricas como parte fundamental na engenharia de software, como deve ser realizada e quais os benefícios que trazem a utilização dessa metodologia no planejamento.

3.1 MÉTRICAS NA GERÊNCIA DE PROJETOS

O uso de métricas tem representado uma ferramenta essencial à gerência de projeto de software. Muitos engenheiros de software medem as características do mesmo a fim de obter uma visão clara de uma série de

aspectos, tais como o atendimento dos produtos de software aos requisitos especificados, desempenho do processo de desenvolvimento, esforço/custo despendido entre outros (VASCONCELOS, 2003).

As medições podem ser úteis sob diferentes óticas. Por exemplo, o gerente de projeto pode utilizar as medições para se certificar se estimativas realizadas no início do projeto foram efetivas, acompanhando o esforço, prazo, custo e tamanho, comparando dados planejados com dados reais. Por outro lado, o cliente pode usar medições para determinar se o mesmo atende aos requisitos e se possui um grau de qualidade satisfatório.

Considerado o “pai da gestão administrativa”, Peter Drucker criou a famosa citação, “O que não se pode medir não se pode gerenciar”. Esta premissa é aplicável ao problema da produtividade no desenvolvimento de software, uma vez que se uma empresa for capaz de identificar os programadores ou equipe mais produtiva e eliminar ou de alguma forma capacitar os menos produtivos, trará grandes benefícios tratando-se da competitividade no mercado (PECK, 2006).

3.2 APERFEIÇOAMENTO DO DESENVOLVIMENTO

O único modo racional para aperfeiçoar qualquer processo é medir os atributos específicos, desenvolvendo um conjunto de métricas significativas, e depois usar as métricas para fornecer indicadores que levarão a uma estratégia de aperfeiçoamento (PRESSMAN, 2005).

Paulish (1994) afirma que o processo é apenas um dos “fatores controláveis para melhoria da qualidade de software e do desempenho da organização”.

Sayão propôs um modelo (SAYÃO, 2004), apresentado na Figura 3-1, que foca o resultado do processo em competências.

FIGURA 3-1 COMPETÊNCIA EM PROCESSO (SAYÃO, 2004).

O fator humano está representado na Figura 3-1 pelas habilidades, Davenport (1994) exemplifica o fator humano como fazer com que a tomada de decisão desça na escala organizacional, incrementando na melhoria de tempo de ciclo do processo.

As máquinas e equipamentos estão representados pelos recursos e a organização pela metodologia de trabalho, pela divisão do trabalho organizado de modo que todo o processo seja supervisionado por uma única pessoa ou grupo de pessoas.

A estes três elementos juntos dá-se o nome de tecnologia. A maior ou menor habilidade, a utilização de recursos modernos ou a melhor organização do processo pode levar a um fator muito importante e que em muitos casos é crucial para a organização, que é o fator custo. Neste caso, o custo está relacionado à competência de execução do processo.

Segundo Pressman (1988), nós medimos a eficácia de um processo de software indiretamente, isto é, originamos um conjunto de métricas, baseadas nas saídas que podem ser derivadas do processo.

As saídas incluem medidas dos erros descobertos antes da entrega do software, defeitos entregues aos usuários finais e por eles relatados, produtos

de trabalho entregues (produtividade), esforço humano despendido, tempo gasto, cumprimento do cronograma e outras medidas.

3.3 MÉTRICAS DE PROJETO

As métricas de projeto de software são táticas, isto é, métricas de projeto e indicadores derivados delas são utilizadas por um gerente de projeto e por uma equipe de software, para adaptar o fluxo de trabalho e as atividades técnicas do projeto. Diferentemente de métricas de processo de software (definido na seção anterior) que são utilizados com a finalidade estratégica.

Uma das aplicações das métricas de projeto ocorre durante a estimativa. As métricas coletadas de projetos anteriores são utilizadas como base, a partir da qual estimativas de esforço e de tempo são feitas para o trabalho atual de desenvolvimento (PRESSMAN, 1988).

Conforme o projeto prossegue, medidas de esforço e de tempo utilizados são comparadas com as estimativas originais (e o cronograma do projeto). O gerente projeto utiliza esses dados para monitorar e controlar o progresso.

Pressman (2005) afirma que o objetivo das métricas de um projeto é duplo. Por um lado, minimizando o cronograma de desenvolvimento, fazendo eventuais ajustes para evitar atrasos e problemas, e riscos em potencial. Por outro lado, as métricas de projetos podem ser utilizadas para avaliar a qualidade do produto durante sua evolução e, se necessário, modificar a abordagem para aperfeiçoar a qualidade.

À medida que a qualidade é aperfeiçoada, os defeitos são minimizados, e conforme a contagem de defeitos decresce, a quantidade de retrabalho durante o projeto é também reduzida. Isso leva a diminuição do custo total do projeto (PAULISH, 1994).

3.4 MÉTRICAS DE SOTWARE

A medição na área de software é um tema discutido há mais de 25 anos. Este tema tem sido tratado principalmente de dois pontos de vista, desde o ponto de vista dos pesquisadores focados na análise da complexidade do software e desde o ponto de vista dos desenvolvedores de software focados na

busca da melhoria dos processos de desenvolvimento (GLASS apud BRABARÁN, 2006).

Nos últimos anos, frente a este ambiente de competitividade e o crescimento das aplicações de software em tamanho e complexidade, a procura por programas que permitam alcançar melhorias nos processos de desenvolvimento, nas indústrias de software tem-se intensificado.

Os Estudos de medição do software tem se orientado principalmente ao estudo da aplicação de métricas de software dentro de um planejamento dos projetos, assim como também a medir sua qualidade e produtividade. Assim, esta informação servirá para a administração e controle do processo de desenvolvimento e para determinar melhorias alcançadas nos resultados (PRESSMAN, 2005; FENTON, 1991 apud FRANCISCHINI, 2005).

No entanto, apesar das métricas de software serem vistas como as ferramentas para avaliação e predições das diferentes atividades do projeto de software, a situação atual da indústria de software revela que o uso de medidas neste setor tem sido limitado. Poucos estudos indicam como um sistema de medição integrado pode apoiar as decisões de gerenciamento.

O melhoramento do processo de gerenciamento dependerá da capacidade para identificar os métodos e parâmetros essenciais de controle de processo de desenvolvimento. Tudo isto pode ser alcançado através do gerenciamento de software mais efetivo o qual por sua vez pode ser facilitado pelo uso de métricas de software mais adequadas (FRANCISCHINI, 2005).

Na pesquisa de avaliação realizada por FRANCISCHINI (2005), como apresenta a Figura 3-2, foi constatado que 63% das indústrias de software não usavam indicadores de desempenho para avaliar o desenvolvimento de projetos de software, sendo que só 8% usavam indicadores sistematicamente.

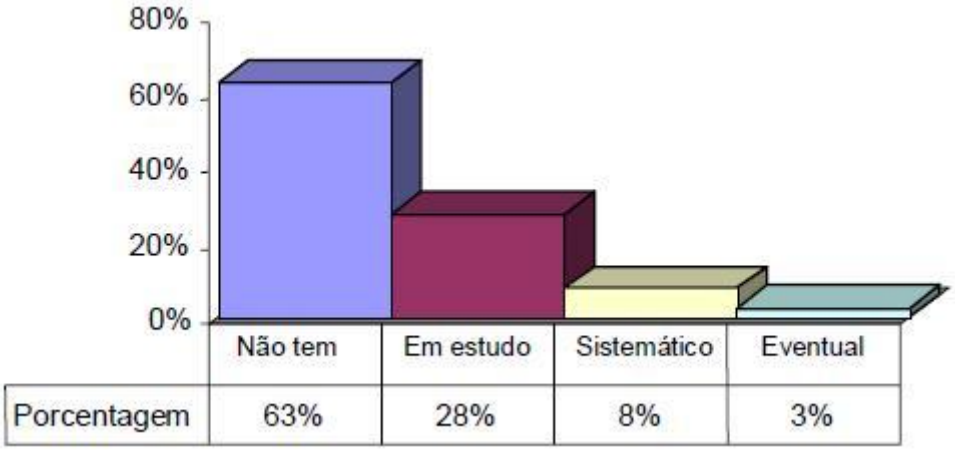


FIGURA 3-2 PORCENTAGEM DE INDÚSTRIAS DE SOFTWARE QUE UTILIZAM SISTEMAS DE INDICADORES PARA AVALIAR O DESEMPENHO DO DESENVOLVIMENTO DE PROJETOS DE SOFTWARE (FRANCISCHINI, 2005).

Com base nos 8% das indústrias que usam métricas, foi identificado que a métrica mais utilizada nestas indústrias é a métrica Pontos de Função (Figura 3-3). O que pode ser confirmado também ao identificar que um dos indicadores mais usados pelas indústrias que possuem indicadores é aquele que relaciona o número de pontos de função sobre o número de homens-hora, como pode ser observado na figura a seguir.

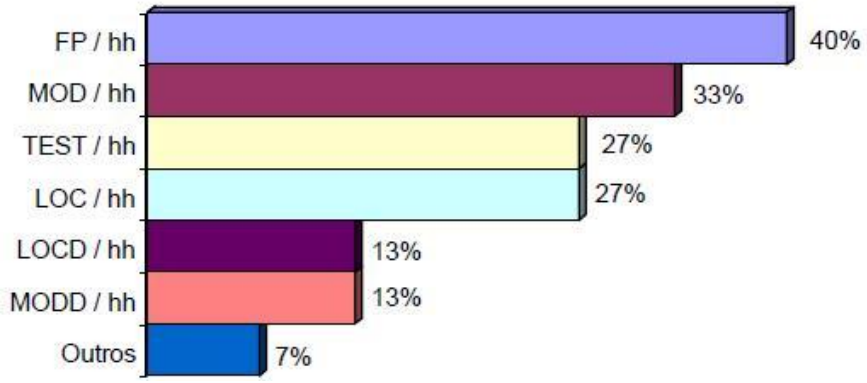


FIGURA 3-3 INDICADORES DE DESEMPENHO MAIS USADOS NA INDÚSTRIA DE SOFTWARE. (FRANCISCHINI, 2005).

Ainda que a maioria das indústrias de software não possua um sistema de indicadores de desempenho, foi verificado que existe uma forte motivação

para elaboração e implantação de um sistema de indicadores pelo desejo e necessidade de desenvolver projeto com melhor qualidade.

A Figura 3-4 mostra as principais razões para a implantação de um sistema de indicadores, que atualmente é visto como uma necessidade no desenvolvimento de projetos de software.

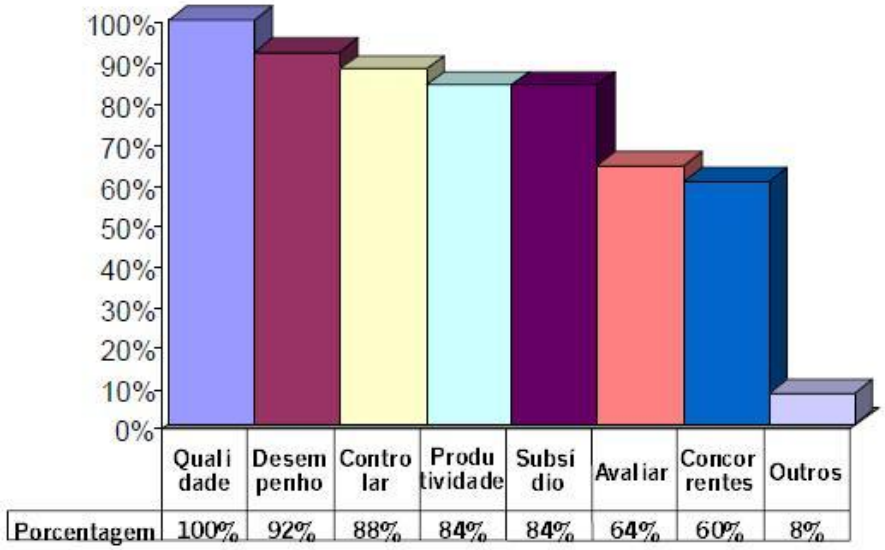


FIGURA 3-4 PRINCIPAIS RAZÕES PARA A IMPLANTAÇÃO DE UM SISTEMA DE INDICADORES. (FRASCISCHINI, 2005).

Com a inclusão das normas de qualidade, foi imposta uma obrigação de implantar um sistema de indicadores de desempenho caso uma empresa deseja ter um certificado de qualidade. Porém considerando a quantidade de empresas que atuam no mercado brasileiro, ainda se verifica a ausência de indicadores em grande parte das empresas de médio a pequeno porte, onde os fatores podem ser observados na Figura 3-5.

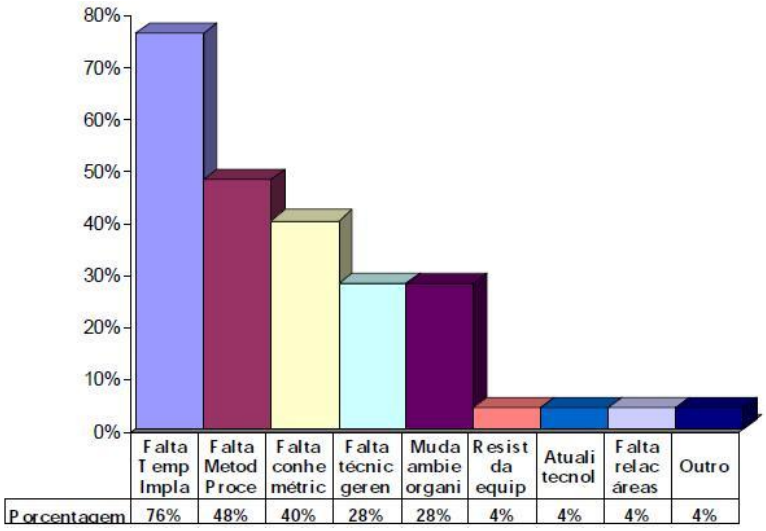


FIGURA 3-5 PRINCIPAIS CAUSAS PARA A AUSÊNCIA DE UM SISTEMA DE INDICADORES NAS INDÚSTRIAS DE SOFTWARE (FRANCISCHINI, 2005).

A principal causa da ausência de um sistema de indicadores nas indústrias de software é a falta de tempo para sua implantação, tempo que é considerado longo demais, pois o processo de implantação teria que começar desde a definição e organização dos projetos.

Como foi citado anteriormente por Drucker, “o que não se pode medir não se poder gerenciar”, e com base nas normas de qualidade, que obrigam a utilização de indicadores para a certificação, torna-se a utilização dos indicadores de desempenho fundamental em uma organização que visa a excelência não só de seus produtos finais, mas também o processo de desenvolvimento do mesmo.

A seguir serão apresentados métodos de medição de software, estes que serão avaliados para a utilização no cálculo da produtividade no processo de desenvolvimento de software.

3.4.1 LINHAS DE CÓDIGO (LOC)

Esta medida é a de maior uso, relaciona o tamanho da saída sobre a quantidade do esforço, onde o tamanho é usualmente medido como as linhas de codificação, o esforço é medido pelo número de pessoas.

Segundo Jones (1986), estas medidas tem apresentado algumas controvérsias, e não são universalmente aceitas como a melhor maneira de se medir o processo de desenvolvimento do software.

A maior parte da controvérsia gira em torno de que as medidas de LOC são dependentes da linguagem de programação, penalizando, portanto, programas bem projetados, porém mais curtos, que elas não podem acomodar facilmente linguagens não procedimentais e que o seu uso em estimativas requer um nível de detalhes que pode ser difícil de conseguir (BARBARÁN, 2002).

3.4.2 PONTOS DE FUNÇÃO (APF)

Esta medida foi desenvolvida por Albrecht (1979) em 1974, quando trabalhava na IBM. O Ponto de Função mede o tamanho do software pela quantificação de sua funcionalidade externa, baseada no projeto lógico ou a partir do modelo de dados.

Para computar o número de pontos de função (FP), primeiro devem-se calcular os pontos de função não ajustados, que são os pontos que abrangem a funcionalidade específica requerida pelo usuário para o projeto e são determinados em base a cinco características do domínio de informação: (1) número de entradas do usuário, (2) número de saída do usuário, (3) número de consultas do usuário, (4) número de arquivos, e (5) número de interfaces externas.

A cada uma destas características é atribuída uma complexidade subjetiva avaliada numa escala ordinal de três pontos: baixa, média, alta.

Segundo deve-se calcular o valor dos pontos de função ajustados, que são características do sistema que avaliam a funcionalidade geral da aplicação, e aos quais se lhes atribui um nível de influência que varia numa escala de 0 a 5.

Finalmente para determinar os pontos de função é preciso multiplicar o valor dos pontos de função não ajustados, pelo valor dos pontos de função ajustados.

Com o resultado desta medição e sabendo-se a produtividade média da organização para produzir um PF, pode-se então estimar o esforço total para o projeto.

3.4.3 PONTOS POR CASO DE USO (PCU)

Os Pontos de Casos de Uso (PCU) foram criados em 1993 por Gustav Karner da empresa Objectory AB. Esta métrica permite fazer estimativas no início do projeto com base no modelo de casos de uso. A filosofia dos Pontos de Casos de Uso é baseada na definição da Análise de Pontos por Função (APF), na qual a funcionalidade vista pelo usuário é a base para a estimativa do tamanho do software (BELGAMO; FABBRI, 2004).

Há alguns anos atrás (início da década de 90) se tinha a falsa noção de que a APF não era adequada para medir sistemas orientados a objetos. Aqueles que compartilhavam desta noção, na prática desconheciam a APF.

Com a disseminação da construção e projeto de sistemas orientados a objetos, houve também uma mudança na forma de se especificar e modelar os sistemas. A UML e os casos de uso rapidamente tornaram-se padrão na indústria de software (FATTO, 2010).

Dentro deste contexto, em 1993 Gustav Karner propôs em um trabalho acadêmico a metodologia dos Pontos por Caso de Uso (baseado na Análise de Pontos de Função) com o intuito de estimar recursos para projetos de software orientados a objeto desenvolvidos utilizando o processo *Objectory.*

1. O processo de medica do PCU consiste resumidamente em:
2. Contar os atores e identificar sua complexidade;
3. Contar os casos de uso e identificar sua complexidade;
4. Calcular os PCU’s não ajustados;
5. Determinar o fator de complexidade ambiental;
6. Calcula os PCU’s ajustados;

Da mesma forma que acontece no método de pontos de função com o resultado desta medição e sabendo-se a produtividade média da organização para produzir um PCU, pode-se então estimar o esforço total para o projeto.

3.4.4 VANTAGENS E DESVANTAGENS

O objetivo do trabalho não é realizar a medição utilizando todas as metodologias expostas, e sim definir qual é a mais conveniente para a empresa do estudo de caso.

Por isso, se faz necessário considerar certos aspectos que supram os requisitos do trabalho, que no caso é realizar a distribuição de esforço pelas fases do projeto de desenvolvimento.

A tabela 3-1 demonstra um comparativo destes aspectos entre as metodologias apresentadas.

Requisitos\Método	SLOC	FPA	UCP
Bibliografia ampla	X	X	
Independência de linguagem		X	X
Lida com múltiplas linguagens		X	X
Ferramentas	X	X	
Aplica-se nas fases iniciais do projeto		X	X[1]
Padronização para contagem		X	X[2]
Padrão ISO		X	
Existência de um grupo de usuários ou organização responsável pela padronização ou evolução do método	X[3]	X	

TABELA 3-1 COMPARATIVO DOS REQUISITOS DAS METODOLOGIAS.

É importante ressaltar alguns aspectos destacados na tabela 3-1:

- X[1]: Devido ao processo de medição do PCU ser baseado em casos de uso, o método não pode ser empregado antes de concluída a análise de requisitos do projeto, além de ser necessário que a empresa o utilize;
- X[2]: Não existe um padrão único para a escrita do caso de uso. Diferentes estilos na escrita dos casos de uso ou na sua granularidade podem levar a resultados diferentes na medição por PCU;
- X[3]: Não foi encontrado nenhum grupo específico, porém existem muitos artigos e ferramentas que utilizam o método.

Dentre todos os requisitos levantados o fato da padronização da contagem e aplicação nas fases iniciais do projeto levaram a escolhe do método de medição por pontos de função para o cálculo da produtividade.

CONSIDERAÇÕES PARCIAIS

O estudo e prática das metodologias de medição é um grande aliado do processo de desenvolvimento de software. Como apresentado anteriormente apesar do baixo índice de utilização, existe grande motivação das empresas em adotarem o conceito na melhoria do processo de desenvolvimento que por sua vez reflete na qualidade do produto final.

Diante das metodologias estudadas, foi visto que o método de pontos de função é o que melhor se adéqua aos requisitos do trabalho para verificar a produtividade da empresa. Dentre uma série de fatores, os principais condizem com a utilização nas fases iniciais do projeto, bem como a padronização existente para que se faça a contagem.

No próximo capítulo será apresentada a empresa em que o estudo de caso foi realizado, a estrutura dos processos internos e como será feito o processo de contagem por pontos de função e os outros dados necessários para indicar a produtividade.

4 ESTUDO DE CASO

4.1 EMPRESA

A PROSYST foi fundada em 15 de outubro de 1986, e está completando 24 anos de atuação no mercado de sistemas de gestão empresarial que funcionam de maneira modularizada, permitindo integração gradativa à medida que são implantados. A informatização abrange as diversas áreas de uma indústria ou organização, onde os sistemas passam a ser verdadeiras soluções integradas para os seus usuários. Seu principal diferencial é o fato de ser uma empresa certificada ISO 9001 desde 20 de julho de 2001 através do organismo certificador BVQI (Bureau Veritas Quality International) (PROSYST, 2006).

Os sistemas foram modelados e construídos com ferramentas de última geração que possibilitam a aplicação das mais modernas tecnologias incluindo integração com a Internet. A atualização tecnológica é uma preocupação constante da PROSYST, que tem investido sistematicamente em ferramentas e treinamento visando o contínuo aperfeiçoamento de seus sistemas. Com uma arquitetura flexível, os sistemas permitem adaptações personalizadas com alto grau de aderência às necessidades do cliente.

Disponíveis nas mais diversas plataformas e ambientes operacionais, os sistemas PROSYST não requerem grandes investimentos em ambiente de informática. A implantação escalável, módulo a módulo, permite que o investimento ocorra na medida da necessidade.

Sua missão é oferecer ao mercado Soluções de Qualidade em Sistemas Informatizados de Gestão Empresarial, que sejam modulares, adaptáveis e integráveis, objetivando aos seus Clientes, a racionalização, o controle e a eficácia no uso destas informações.

4.1.1 PRODUTOS

Os produtos desenvolvidos e comercializados pela PROSYST são: Sistemas de Gestão Empresarial; Customizações dos sistemas (serviços de desenvolvimento e adaptações dos sistemas às necessidades do cliente);

Serviços de manutenção e suporte dos sistemas desenvolvidos pela PROSYST.

A PROSYST desenvolve e estabelece os processos necessários para a realização dos produtos. No planejamento da realização dos produtos são determinados: objetivos da qualidade através das metas estabelecidas para os indicadores estatísticos; requisitos para o produto através do levantamento de informações e definições das características conforme atividades do processo de Desenvolvimento de Sistemas ou das descrições das Ordens de Serviços; os processos para a realização, verificação, validação e monitoramento dos Sistemas e Serviços estão estabelecidos e documentados através do processo de Desenvolvimento de Sistemas e do processo de Suporte Técnico; e registros da qualidade que são gerados para evidenciar que o processo de realização e o produto resultante atendem aos requisitos que estão definidos nos procedimentos correspondentes.

A PROSYST preserva a conformidade dos seus produtos durante o processo interno até a entrega ao cliente. Esta preservação inclui a identificação das versões dos programas, cuja integridade é mantida ao longo do seu desenvolvimento através da realização de cópias de segurança (backup) dos fontes.

Enfim, a PROSYST monitora as características dos sistemas para verificar se os seus requisitos estão sendo atendidos. Esse monitoramento é realizado através de testes no Sistema Controle de Serviços, onde são mantidos registros que fornecem evidências de que os produtos foram testados. Os registros identificam a pessoa responsável pela liberação do produto. Os sistemas não são liberados aos clientes até que o monitoramento necessário seja realizado, a menos que seja aprovado de outra maneira, pela diretoria técnica e quando aplicável, pelo cliente.

4.1.2 ESTRUTURA

O sistema de gestão empresarial PROSYST (ERP-PROSYST) é composto de módulos integrados que atendem às áreas de:

- Suprimentos: administram de forma eficiente as compras com total controle sobre os preços através de metas, associados ao prazo de atendimento e qualidade;
- Vendas: emitem orçamentos, possuem aprovação de pedidos, efetuam análise de crédito do cliente, permitem efetuar recebimentos financeiros antecipados e emitem pedidos de vendas personalizados;
- Custos e Preço de Venda: permitem identificar a composição de materiais e processos a qualquer tempo e sem limite de níveis e também a emissão de informações relativas à análise de rentabilidade;
- Controladoria: integram as informações de todos os módulos, dispensando a digitação manual de lançamentos contábeis. O usuário pode criar relatórios configuráveis, com liberdade para composição dos valores. Emitem os livros e disponibilizam os registros históricos;
- Fiscal: geram os livros fiscais de entrada, saída e guia eletrônica, automatizando todos os processos de apuração. Possibilitam e emissão de dados para a Receita Federal e Receita Estadual;
- Financeiro: administram os pagamentos e recebimentos antecipados. Efetuam programação e liberação de pagamentos e integram-se a diversos bancos;
- Inflação Interna: auxiliam no controle da Inflação Interna, permitindo que seja feito um acompanhamento minucioso da evolução dos custos diretos e indiretos de fabricação;
- Controle de Industrialização de Produtos: efetuam o controle dos produtos que estão sendo industrializados por terceiros, bem como os produtos de terceiros que estão sendo industrializados na empresa; e
- Produção: acompanham, controlam e, se necessário, rastreiam até a origem todas as atividades que transformam ou transformaram materiais e componentes em produtos acabados.

Portanto, pode-se perceber que os módulos do sistema de gestão empresarial PROSYST abrangem a informatização de diferentes áreas de uma empresa, com total integração dos dados. E por se tratarem de sistemas modularizados, sua implantação pode ser gradativa, de acordo com as necessidades do cliente.

4.2 INDICADORES DA EMPRESA

A PROSYST conta atualmente com um sistema de indicadores, padronizado pela certificação ISO 9001.

A Tabela 4-1 mostra os indicadores por processo atuante na empresa.

Processo	**Indicadores de Monitoramento**	**Indicadores de Medição**
Comercial	Contratos Vigentes	Avaliação Satisfação dos Clientes (Empresa) - Gráfico Avaliaçao da Satisfacao dos Clientes (Atend. Adm) - Gráfico
Desenvolvimento de Sistemas	Atraso Médio das OS´s	Evolução de Sistemas Cumprimento dos Prazos de Entrega - Gráfico Cumprimento dos Prazos de Entrega - Relatório Avaliação Satisfação dos Clientes (Sistemas) – Gráfico Avaliação Satisfação dos Clientes (Projetos) - Gráfico
Suporte Técnico	Gráfico de Acompanhamento de Chamados	Chamados em Aberto (Média Mensal) - Gráfico Avaliação Satisfação Clientes (Suporte) - Gráfico
Gestão Administrativa	Resumo PPR (ano vigente)	Avaliação Satisfação Clientes (Empresa) - Gráfico Avaliação Satisfação Clientes (Sistemas) - Gráfico Avaliação Satisfação Clientes (Suporte) - Gráfico Avaliaçao da Satisfacao dos Clientes (Atend. Adm) - Gráfico Avaliação Satisfação Clientes (Média Geral) - Gráfico

TABELA 4-1 INDICADORES DA EMPRESA.

O processo relacionado ao trabalho é a do setor de desenvolvimento de sistemas, que consiste em uma equipe de consultores, analistas e programadores.

A Tabela 4-2 mostra o objetivo do processo para execução.

Objetivo	Coordenação do Processo
Executar as ordens de serviços que atendam aos requisitos dos clientes dentro dos prazos acordados.	**DESENVOLVIMENTO**

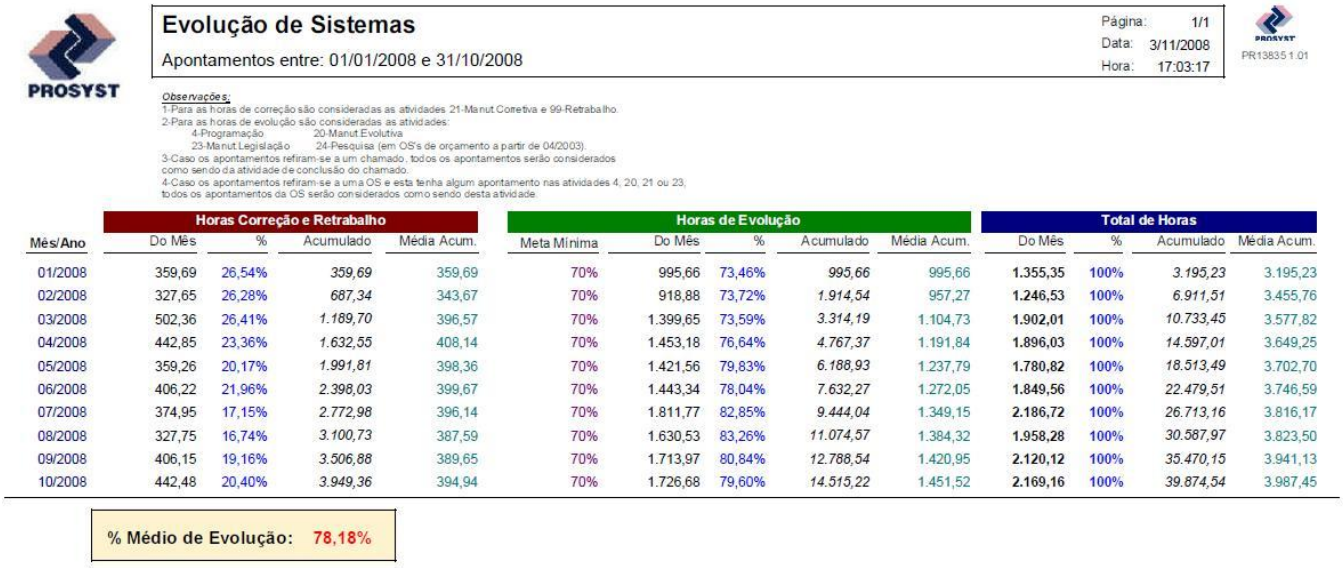
Entradas do Processo	
De qual processo?	O que fornece?
Gestão Administrativa	Política da Qualidade e objetivos; Ações de Melhoria; Resultados do Sistema de Gestão da Qualidade; Recursos liberados.
Gestão da Qualidade	Programa de auditorias internas; Resultados das auditorias realizadas.
Comercial	Orçamentos enviados; Ordens de Serviços;
Suporte Técnico	Ordens de Serviços a partir de chamados.
Infra-estrutura	Realização de manutenções; Restauração de dados do backup.

TABELA 4-2 OBJETIVO DO PROCESSO.

Como foi visto na Tabela 4-1, o processo de desenvolvimento possui 4 indicadores de medição, são eles:

- Evolução de Sistemas;

- Cumprimento dos Prazos de Entrega;
- Avaliação Satisfação dos Clientes (Sistemas).
- Avaliação Satisfação dos Clientes (Projetos);



PROSYST

Evolução de Sistemas

Apontamentos entre: 01/01/2008 e 31/10/2008

Página: 1/1
Data: 3/11/2008
Hora: 17:03:17

PR13835 1.01

Observações:
1-Para as horas de correção são consideradas as atividades 21-Manut.Corretiva e 99-Retrabalho.
2-Para as horas de evolução são consideradas as atividades:
4-Programação 20-Manut.Evolutiva
23-Manut.Legislação 24-Pesquisa (em OS's de orçamento a partir de 04/2003).
3-Caso os apontamentos refiram-se a um chamado, todos os apontamentos serão considerados como sendo da atividade de conclusão do chamado.
4-Caso os apontamentos refiram-se a uma OS e esta tenha algum apontamento nas atividades 4, 20, 21 ou 23, todos os apontamentos da OS serão considerados como sendo desta atividade.

	Horas Correção e Retrabalho				Horas de Evolução					Total de Horas			
Mês/Ano	Do Mês	%	Acumulado	Média Acum.	Meta Mínima	Do Mês	%	Acumulado	Média Acum.	Do Mês	%	Acumulado	Média Acum.
01/2008	359,69	26,54%	*359,69*	359,69	70%	995,66	73,46%	*995,66*	995,66	**1.355,35**	100%	*3.195,23*	3.195,23
02/2008	327,65	26,28%	*687,34*	343,67	70%	918,88	73,72%	*1.914,54*	957,27	**1.246,53**	100%	*6.911,51*	3.455,76
03/2008	502,36	26,41%	*1.189,70*	396,57	70%	1.399,65	73,59%	*3.314,19*	1.104,73	**1.902,01**	100%	*10.733,45*	3.577,82
04/2008	442,85	23,36%	*1.632,55*	408,14	70%	1.453,18	76,64%	*4.767,37*	1.191,84	**1.896,03**	100%	*14.597,01*	3.649,25
05/2008	359,26	20,17%	*1.991,81*	398,36	70%	1.421,56	79,83%	*6.188,93*	1.237,79	**1.780,82**	100%	*18.513,49*	3.702,70
06/2008	406,22	21,96%	*2.398,03*	399,67	70%	1.443,34	78,04%	*7.632,27*	1.272,05	**1.849,56**	100%	*22.479,51*	3.746,59
07/2008	374,95	17,15%	*2.772,98*	396,14	70%	1.811,77	82,85%	*9.444,04*	1.349,15	**2.186,72**	100%	*26.713,16*	3.816,17
08/2008	327,75	16,74%	*3.100,73*	387,59	70%	1.630,53	83,26%	*11.074,57*	1.384,32	**1.958,28**	100%	*30.587,97*	3.823,50
09/2008	406,15	19,16%	*3.506,88*	389,65	70%	1.713,97	80,84%	*12.788,54*	1.420,95	**2.120,12**	100%	*35.470,15*	3.941,13
10/2008	442,48	20,40%	*3.949,36*	394,94	70%	1.726,68	79,60%	*14.515,22*	1.451,52	**2.169,16**	100%	*39.874,54*	3.987,45

% Médio de Evolução: 78,18%

FIGURA 4-1 RELATÓRIO DO INDICADOR DE MEDIÇÃO EVOLUÇÃO DE SISTEMAS.

O indicador de evolução de sistemas é baseado no apontamento de horas do funcionário, onde o mesmo, no ato de apontar suas horas (Figura 4-2) terá de indicar atividade exercida no intervalo de horas. Esta atividade por sua vez tem um vínculo com a Ordem de Serviço, a mesma tendo dois tipos:

- Manutenção: que consiste em horas não faturáveis (horas ruim), onde o apontamento contabiliza para as horas de correção e retrabalho.
- Orçamento: consiste nas horas faturáveis (horas boas), onde poderá estar sendo realizado um projeto para programas novos ou atualização a pedido do cliente.

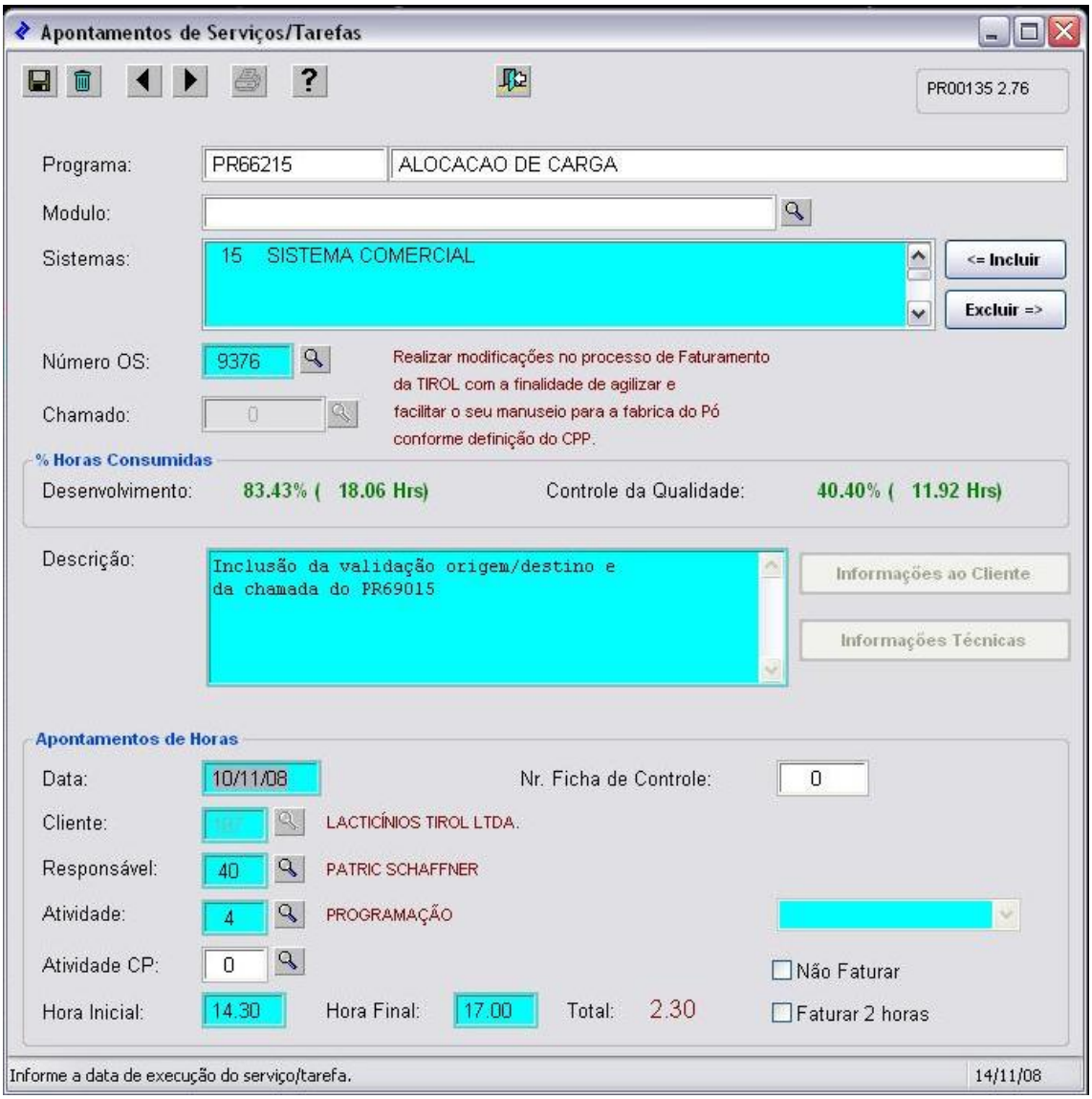


FIGURA 4-2 PROGRAMA DE APONTAMENTO DE HORAS.

Com base neste indicador os gerentes de processo podem ter a noção do porque a empresa não está obtendo o lucro planejado, por exemplo, porém este controle se torna incompleto a partir do momento que existe a necessidade de melhoramento do processo, visto através destes indicadores não se sabe em qual dos processos de desenvolvimento está o(s) problemas.

CONSIDERAÇÕES PARCIAIS

Como foi verificado, a empresa PROSYST não possui um indicador de produtividade, que como foi comentado anteriormente, é de suma importância para otimizar o desenvolvimento de processos.

A motivação deste trabalho está na necessidade de se ter uma melhor estimativa na distribuição de horas pelas fases do projeto, bem como o total de horas que deverá ser utilizado pela equipe para realização do mesmo. De forma que não haja prejuízo além de adotar um padrão para estimativas.

O próximo capítulo descreve de forma mais detalhada como será realizado este processo de medição e obtenção da produtividade da empresa.

5 PROCESSO DE MEDIÇÃO

Para realizar a medição a empresa PROSYST se dispôs a oferecer os dados e códigos fonte (medição) necessários para o processo, porém foi requisitado a não inclusão de códigos fonte no conteúdo do trabalho por questões legais e de sigilo. Portanto, a pesquisa, medição e análises serão resultantes de dados e fatores condizentes com a realidade da empresa.

Na seção de anexos é possível ter acesso ao “Controle de Projeto” que consiste nas especificações do projeto que serviram como base para realizar a contagem.

5.1 METODOLOGIA

Como foi visto nas seções anteriores, existem diferentes formas de se medir um software. Para este trabalho, a proposta é utilizar o método por pontos de função (APF).

A metodologia utilizada para medição segue os padrões da IFPUG, através do livro Análise de Pontos de Função (VAZQUEZ, 2003).

A figura 5-1 apresenta o processo de contagem:

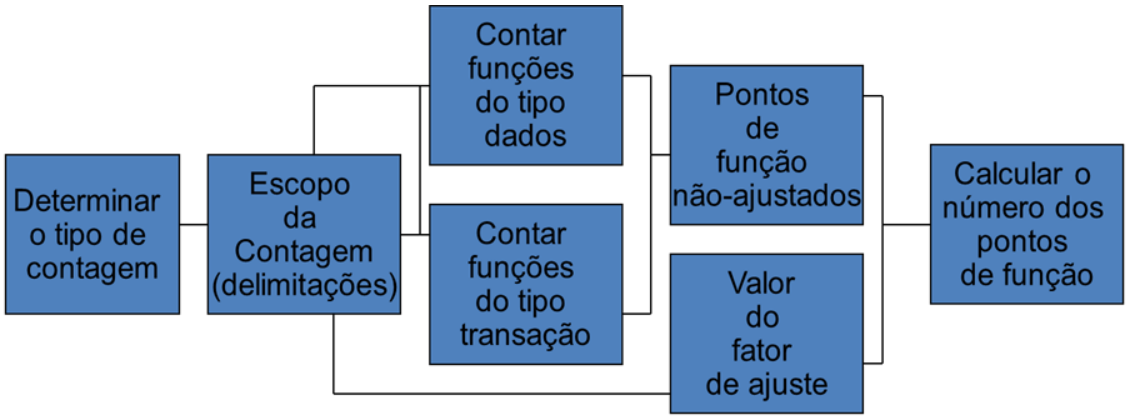


FIGURA 5-1 VISÃO GERAL DO PROCESSO DE CONTAGEM DE PONTOS DE FUNÇÃO (VAZQUEZ, 2003).

Nas próximas seções cada uma das etapas será brevemente explicada, além da apresentação dos resultados obtidos durante o processo para o projeto do estudo de caso.

5.1.1 TIPO DE CONTAGEM

A metodologia prevê três tipos de contagem (IFPUG, 2010):

- Projeto de desenvolvimento;
- Projeto de melhoria;
- Aplicação.

A proposta do trabalho é realizar a medição de um projeto de desenvolvimento: Recebimento do leite (ANEXO A).

5.1.2 ESCOPO DA CONTAGEM

O escopo da contagem consiste nos programas que foram desenvolvidos no projeto (ANEXO A).

A figura apresenta a lista de programa em qual serão aplicados a metodologia.

Programa	Descrição	Linguagem
PR00112	Parâmetros do Recebimento do Leite	COBOL
PR00212	Entrada do Leite do Produtor	COBOL
PR00312	Classificação do Leite	COBOL
PR00412	Manutenção das Análises do Leite	COBOL
PR00512	Entrada do Leite no Posto de Resfriamento\Fábrica	COBOL
PR00612	Consulta Recebimento do Leite por Produtor Rural	COBOL
PR00712	Aprovação Cadastro Produtor Rural	COBOL
PR00812	Importação\Exportação de Dados para o Sistema do Leite	COBOL
PR00912	Ajustes no Recebimento do Leite	COBOL
PR01012	Relatórios e Consultas	COBOL
PR01112	Saída do Leite do Posto de Resfriamento	COBOL

FIGURA 5-2 LISTA DOS PROGRAMAS AVALIADOS.

5.1.3 FUNÇÕES DO TIPO DADOS

As funções do tipo dados representam as funcionalidades fornecidas pelo sistema ao usuário, para atender as suas necessidades de dados. São classificadas em (VAZQUEZ, 2003):

- **Arquivo Lógico Interno (ALI):** grupo de dados ou informação de controle logicamente relacionados, reconhecido pelo usuário, mantido dentro da fronteira da aplicação. Sua principal intenção é armazenar dados mantidos por um ou mais processos elementares da aplicação sendo medida.

- **Arquivo de Interface Externa (AIE):** grupo de dados ou informação de controle logicamente relacionados, reconhecido pelo usuário, referenciado pela aplicação, mas mantido dentro da fronteira de outra aplicação. Sua principal intenção é armazenar dados referenciados por um ou mais processos elementares da aplicação sendo contada.

A tabela 5-1 apresenta a contagem para as tabelas envolvidas no projeto e a contribuição do número de pontos de função total.

Tabela	Descrição	TR	TD	Complexidade	Pontos Função
AIE					
CLIENTEM	Cadastro de cliente/fornecedor/produtor	2	4	Baixa	5
FMATER	Cadastro de produtos e materiais	2	5	Baixa	5
FILIAL	Cadastro de filiais	2	3	Baixa	5
MOTORA	Cadastro de motoristas	1	2	Baixa	5
PARAMSIS	Cadastro de parâmetros do sistema	1	3	Baixa	5
ALI					
LEITEACR	Análises do Leite - Códigos Referências	2	6	Baixa	5
LEITEAM	Análises do Leite - Mestre	1	19	Baixa	5
LEITEAPQ	Análises do Leite - Pontuação da Qualidade do Leite	2	6	Baixa	5
LEITECLD	Classificação do Leite - Valores Referências das Análises	2	4	Baixa	5
LEITECLM	Classificação do Leite - Mestre	1	4	Baixa	5
LEITEPRD	Recebimento do Leite do Produto Rural - Detalhes	3	7	Baixa	5
LEITEPRJ	Recebimento do Leite do Produto Rural - Justificativa de Ajustes	3	4	Baixa	5
LEITEPRM	Recebimento do Leite do Produto Rural - Mestre	1	12	Baixa	5
LEITERA	Recebimento e Analise do Leite - Análises	3	8	Baixa	5
LEITERD	Recebimento e Analise do Leite - Detalhes	2	6	Baixa	5
LEITERM	Recebimento e Analise do Leite - Mestre	1	20	Média	10
TOTAL		**29**	**113**		**85**

TABELA 5-1 CONTAGEM DAS FUNÇÕES DO TIPO DADOS.

Os tipos de registros (TR) são subconjuntos de dados dentro de um ALI/AIE, que foram reconhecidos pelo usuário. Os tipos de dados (TD), por sua vez, são campos reconhecidos pelo usuário como único e não repetido.

Utilizando-se desses dados, pode-se chegar à sua complexidade, utilizando a tabela 5-2.

TR\TD	**1 a 19**	**20 a 50**	**51 ou mais**
1	baixa	baixa	média
2 a 5	baixa	média	alta
6 ou mais	média	alta	alta

TABELA 5-2 TABELA DE COMPLEXIDADE PARA FUNÇÕES DE DADOS.

5.1.4 FUNÇÕES DO TIPO TRANSAÇÃO

As funções do tipo transação representam as funcionalidades de processamento de dados fornecidas pelo sistema ao usuário. São classificados em:

Entrada Externa (EE): Processa dados ou informações de controle vindos de fora da fronteira da aplicação e cuja intenção é manter um ou mais ALI's e/ou alterar o comportamento do sistema;

Saída Externa (SE): Enviar dados ou informações de controle para fora da fronteira da aplicação. Sua lógica de processamento deve conter formula matemática ou calculo, ou criar dados derivados, manter um ou mais ALI e/ou alterar o comportamento do sistema;

Consulta Externa (CE): Enviar dados ou informações de controle para fora da fronteira da aplicação pela simples recuperação de dados de ALI e/ou AIE. Sua logica de processamento não deve conter formula matemática ou cálculo, nem criar dados derivados, nem manter um ou mais ALI, nem alterar o comportamento do sistema.

A tabela 5-3 apresenta os resultados obtidos na contagem dos programas e relatórios desenvolvidos no projeto.

Programa	Descrição		EE			CE			SE		Pontos Função
		Baixa	Média	Alta	Baixa	Média	Alta	Baixa	Média	Alta	
PR00112	Parâmetros do Recebimento do Leite	1				1					7
PR00212	Entrada do Leite do Produtor		1				1			1	17
PR00312	Classificação do Leite	1			1	1					10
PR00412	Manutenção das Análises do Leite			2			1		1		23
PR00512	Entrada do Leite no Posto de Resfriamento\Fábrica	1		4			4				51
PR00612	Consulta Recebimento do Leite por Produtor Rural					1					4
PR00712	Aprovação Cadastro Produtor Rural	1				1					7
PR00812	Importação\Exportação de Dados para o Sistema do Leite			1							6
PR00912	Ajustes no Recebimento do Leite	1				1					7
PR01012	Relatórios e Consultas							1	3	3	40
PR01112	Saída do Leite do Posto de Resfriamento		1				1				10
TOTAL		5	2	7	1	5	7	1	4	4	182

TABELA 5-3 CONTAGEM DAS FUNÇÕES DO TIPO TRANSAÇÃO.

No cálculo das funções transacionais, semelhante as funções de transição, são considerados dois atributos, os arquivo referenciados, que são os ALI / AIE utilizados na funcionalidade e os tipos de dados que são os dados inseridos, apresentados ou consultados. O ANEXO E apresenta um exemplo da verificação de um programa para achar as funções.

As tabelas (5-4 e 5-5) apresentam como se obter a complexidade das funções transacionais.

AR\TD	1 a 4	5 a 15	16 ou mais
0 ou 1	baixa	baixa	média
2	baixa	média	alta
3 ou mais	média	alta	alta

TABELA 5-4 COMPLEXIDADE ENTRADA EXTERNA.

AR\TD	1 a 5	6 a 19	20 ou mais
0 ou 1	baixa	baixa	média
2 ou 3	baixa	média	alta
4 ou mais	média	alta	alta

TABELA 5-5 COMPLEXIDADE SAÍDA E CONSULTAS EXTERNAS.

5.1.5 PONTOS DE FUNÇÃO NÃO AJUSTADOS

A contagem dos pontos de função não ajustados consiste na soma da contribuição das funções do tipo dados e das funções do tipo transação como apresentado na tabela 5-6.

Totais	PF
Função do tipo dados	85
Função do tipo transação	182
Pontos função não-ajustado	**267**

TABELA 5-6 TOTAL DOS PONTOS DE FUNÇÃO NÃO AJUSTADOS.

Esta soma se dá através da seguinte expressão:

$$NPFi = \sum_{j=1}^{3} NCij \; x \; Cij$$

Onde NCij = número de funções do tipo i (i variando de 1 a 5, segundos os tipos de função existentes: ALI, AIE, EE, SE, CE) que foram classificados na complexidade j (j variando de 1 a 3, segundo os valores de complexidade: baixa, média e alta).

Cij = valor da contribuição da complexidade j no cálculo dos pontos da função i, dado pela tabela 5-6.

5.1.6 DETERMINAÇÃO DO VALOR DO FATOR DE AJUSTE

Cada tipo de contagem (desenvolvimento, melhoria e aplicação) possui uma fórmula específica para a determinação dos pontos de função ajustados.

Tem como objetivo indicar a funcionalidade geral fornecida pela aplicação ao usuário. Calculado com base em 14 características gerais do sistema (CGS), produzindo uma variação de +/- 35% no tamanho.

As características gerais são apresentadas na tabela 5-7 juntamente a pontuação identificada referente ao projeto.

Valor do Fator de Ajuste	Nível de Influência
Comunicação de Dados	2
Processamento Distribuído	1
Performance	4
Configuração Altamente Utilizada	0
Volume de Transações	2
Entrada de Dados-Online	4
Eficiência do Usuário Final	5
Atualização On-Line	3
Comlpexidade de Processamento	3
Reusabilidade	4
Facilidade de Instalação	0
Facilidade de Operação	0
Múltiplos Locais	2
Facilidade de Mudanças	4
TOTAL	34

TABELA 5-7 DETERMINAÇÃO DO VALOR DE AJUSTE.

O cálculo do fator de ajuste é obtido pela formula abaixo:

$$VFA = TNI\ x\ 0{,}01 + 0{,}65$$

onde VAF é o valor do fator de ajuste; e TNI é o total do nível de influência.

5.1.7 CÁLCULO DO NÚMERO DE PONTOS DE FUNÇÃO

O cálculo do número dos pontos de função é obtido na multiplicação do total de pontos não ajustados e o valor do fator de ajuste (VAF) conforme ilustrado na tabela 5-8.

Totais	PF
PF não ajustados	267
Fator de Ajuste	0,99
Pontos Função	**264,33**

TABELA 5-8 NÚMERO DOS PONTOS DE FUNÇÃO.

O total do número de pontos de função é dado pelo somatório dos pontos das tabelas de função multiplicando o valor de cada tipo de função pelo fator de ajuste, encontrado no item 5.1.6:

$$PFNA = \sum_{i=1}^{5} NPFi$$

Sendo que i varia de 1 a 5, segundo os tipos de funções existentes.

5.2 PRODUTIVIDADE

A produtividade será medida na razão entre o esforço (horas) despendido no projeto e o número dos pontos de função calculados no projeto (apresentado nas seções anteriores) pela expressão:

$$P = \frac{H}{PF}$$

Para obter o esforço do projeto foi gerado um relatório (ANEXO B) com todos os apontamentos da equipe durante o projeto, este fornecendo diversos dados (horas por atividade, horas por programador, horas por programa entre outros), inclusive o total geral de horas utilizadas.

Com estas informações chega-se a produtividade da equipe para o projeto de recebimento do leite (ANEXO A) conforme apresentado na tabela 5-9:

	Horas	Produtividade (PF/H)	Produtividade (H/PF)
Planejado	531	**0,498**	2,009
Realizado	938,4	**0,282**	3,550
Diferença (P/R)	43,41%		

TABELA 5-9 CÁLCULO DA PRODUTIVIDADE DO PROJETO.

Como pode ser visto na tabela 5-9 apresenta informações do que foi estimado no projeto sem a utilização de qualquer métrica, conhecida como método de estimativa pela experiência do analista. O total de horas planejadas está muito abaixo do total realizado, concluindo que a estimativa utilizada pela empresa não está de acordo com o "potencial" da equipe.

Com essas informações é possível estimar as horas de um novo projeto, utilizando o cálculo dos pontos de função e a produtividade, onde multiplicando a produtividade obtida e o total de pontos de função do novo projeto tem-se o total de horas estimadas.

É importante ressaltar que em diferentes projetos podem-se encontrar diferentes produtividades. Entre os fatores envolvidos vale destacar o nível de experiência do desenvolvedor além do nível de maturidade no cálculo dos pontos de função do projeto. As diferentes produtividades encontradas servirão como base para o refinamento e com a utilização em outros projetos, chegar o mais próximo do valor ideal para realizar as estimativas mais precisas.

Por fim é possível determinar o cronograma do projeto a partir do esforço estimado. Uma vez determinado o esforço em horas, necessário para a realização de uma determinada atividade, para obter sua estimativa de duração, basta dividir esse valor pelo número de horas trabalhadas pela equipe alocada (VAZQUEZ, 2003).

$$Prazo = \frac{esforço}{recursos}$$

Ou seja, o prazo previsto será a razão entre o esforço previsto e a quantidade de recursos alocada na execução do projeto.

CONSIDERAÇÕES PARCIAIS

A partir do levantamento de informações realizado na empresa, a proposta do trabalho se torna útil para o conhecimento da produtividade no desenvolvimento de software da empresa, coisa que até o momento era desconhecida.

Após a análise realizada no projeto escolhido foi possível chegar a um primeiro índice de produtividade da empresa, este que já pode auxiliar imediatamente na estimativa de um novo projeto aliado a metodologia de medição por pontos de função.

A empresa possui um controle eficiente de apontamentos das horas e com uma análise mais aprofundada pode identificar onde estão os problemas no desenvolvimento de cada projeto, além de que, esses dados serão importantes para a próxima etapa do projeto que consiste na modelagem e desenvolvimento de uma ferramenta para distribuição de esforço pelas fases do projeto visando a melhoria das estimativas de esforço no desenvolvimento.

Nas próximas seções serão apresentados conceitos e modelagem para a criação desta ferramenta.

6 FERRAMENTA DE DISTRIBUIÇÃO DE ESFORÇO

Como foi apresentado anteriormente, o objetivo do trabalho é a melhoria nas estimativas de horas dos projetos de desenvolvimento da empresa PROSYST. Para isso, foram utilizados os conceitos de métricas de software para a medição do tamanho funcional dos programas de um projeto aliada as horas utilizadas no projeto obtendo-se a produtividade de desenvolvimento.

A partir da produtividade é possível realizar novas estimativas para novos projetos utilizando o cálculo dos pontos de função.

Com as informações de controle já existentes na empresa, é possível identificar todos os processos de desenvolvimento que são executados ao longo do projeto. Estes dados são inseridos no sistema pelos funcionários em seus respectivos apontamentos de horas, que utilizam como parâmetros principais o código do projeto, do programa e a atividade (análise, desenvolvimento, testes, etc).

A proposta final é a criação de uma ferramenta, que dada à produtividade e o número dos pontos de função do projeto não somente uma estimativa do total de horas para o projeto, mas também o esforço para cada uma das fases que compõem o seu desenvolvimento. De forma geral, a ferramenta apresentará a quantidade horas estimadas para cada fase do projeto com base na produtividade e número de pontos de função.

Nas próximas seções será apresentado com mais detalhes o funcionamento desta ferramenta.

6.1 CONCEITO DA DISTRIBUIÇÃO

A distribuição das horas pelas fases do projeto é adaptada ao modelo iterativo de ciclo de vida de desenvolvimento de software. A empresa do estudo de caso adota o modelo em cascata o que impede a utilização a risca do conceito.

Por este fato, foi realizada uma adaptação para o processo da empresa, ou seja, foram levantadas as atividades exercidas (ANEXO B) durante o projeto e agrupadas em 5 fases:

- ANÁLISE;
- DESENVOLVIMENTO;
- DOCUMENTAÇÃO;
- TESTES \ CONTROLE DE QUALIDADE;
- IMPLANTAÇÃO.

O sistema da empresa é baseado em *windows forms*, de forma que para cada funcionalidade do sistema em geral é criado um programa.

6.2 PROTÓTIPO

Antes da modelagem e desenvolvimento da ferramenta, foi elaborado um protótipo no formato de planilha com a utilização do Microsoft Excel. Este será utilizado como base para o detalhamento do funcionamento e dados apresentados pela aplicação.

6.2.1 ENTRADA DE DADOS

A ferramenta tem basicamente seis parâmetros de entrada do usuário:

- A produtividade inicial da aplicação é dada pelo usuário, depois atualizada automaticamente com base na média das produtividades dos projetos cadastrados na aplicação. A produtividade de cada projeto finalizado será obtida com a inserção do total de esforço real;
- A produtividade que será na realidade a média de todos os projetos que foram realizados.
- As fases do projeto. Para que a aplicação não seja restrita aos tipos de fases identificados, a aplicação vai permitir ao usuário cadastrar as fases do processo;
- A porcentagem média de horas utilizadas para cada uma das fases incluídas;
- Os programas que serão desenvolvidos;

- O número de pontos de função dos programas do projeto, que são calculados na etapa de análise.

6.2.2 DISTRIBUIÇÃO DE HORAS POR FASE

A tabela de distribuição de horas apresenta as fases do projeto e o número de horas calculadas para cada uma delas conforme a tabela 6-1.

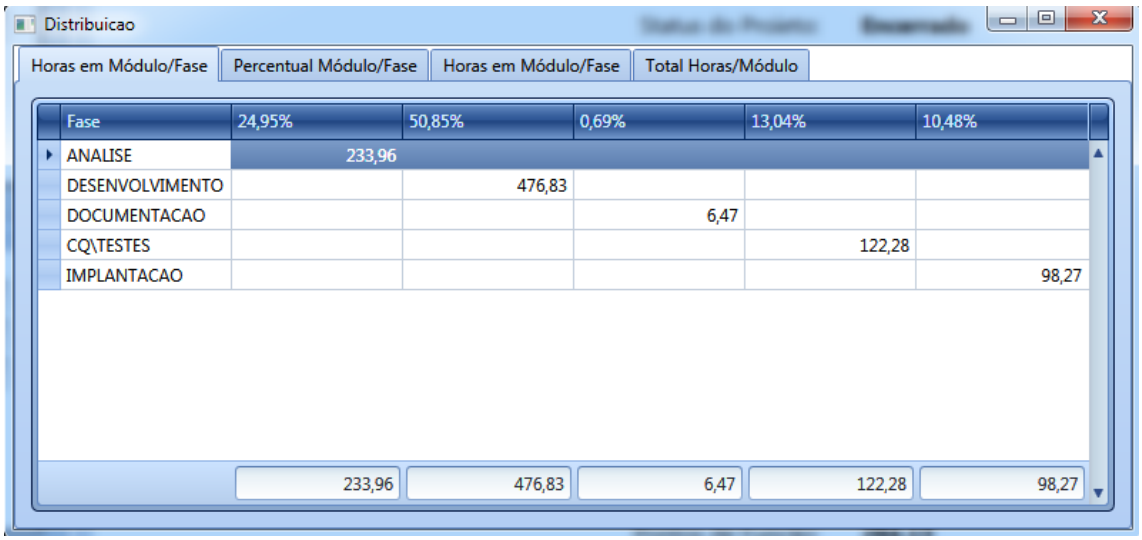


TABELA 6-1 TABELA DE DISTRIBUIÇÃO DE HORAS POR FASE.

O número de horas por fase é obtida a partir das porcentagens cadastras nas fases do projeto a partir da expressão:

$$\mathrm{Hf} = \mathrm{PCf} \times \mathrm{Ht}$$

onde Hf = número de horas da fase, PCf = percentual atribuído a fase, Ht = total de horas estimadas do projeto.

Para chegar aos percentuais é realizada uma análise das atividades executadas em projetos anteriores.

A tabela 6-2 exemplifica o processo de agrupamento e cálculo do percentual. As horas de cada atividade são associadas a uma fase (cores) do projeto.

Assessoria na Sede	Acesso Remoto	Controle de Qualidade	Desenvolvimento	Documentação	Formação Profissional	Testes
134,25	3,17	66,1	466,72	6,5	2,08	56,24

Implantação	Levantamento Informações	Montagem da Proposta	Pesquisa	Reuniões	Suporte Cliente	Total
92,17	65,5	21,84	8,33	12,5	3	**938,4**

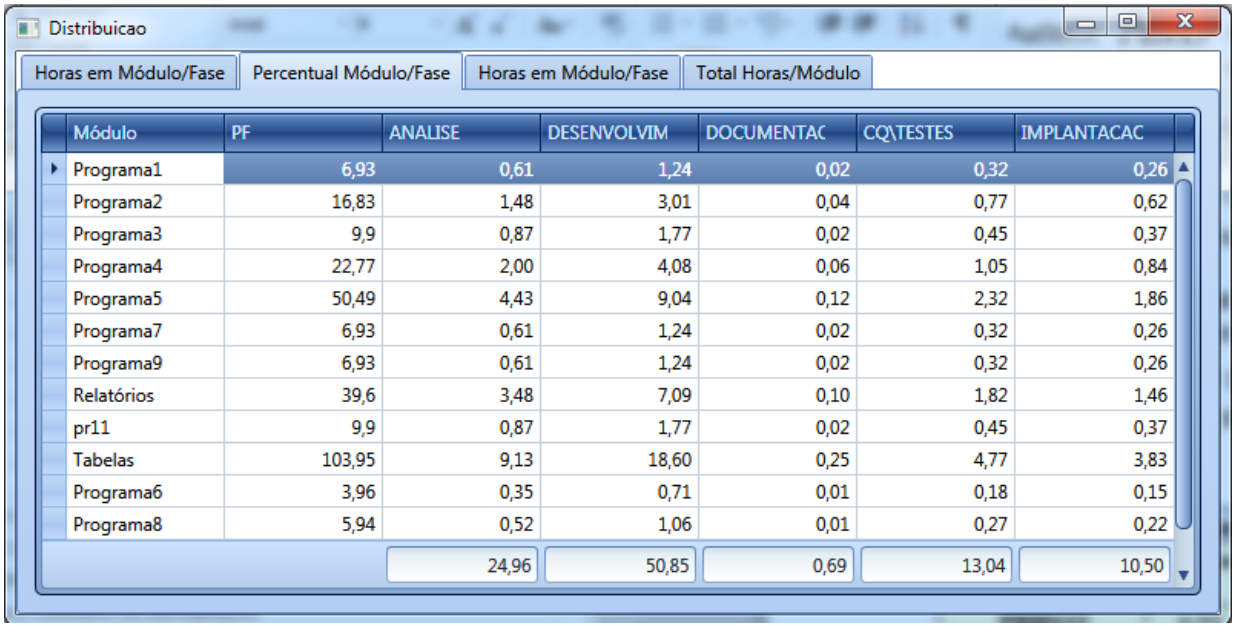
FASE	HORAS	(%)
ANÁLISE	234,09	24,95%
DESENVOLVIMENTO	477,13	50,85%
DOCUMENTAÇÃO	6,5	0,69%
TESTES \ CQ	122,34	13,04%
IMPLANTAÇÃO	95,34	10,16%
MANUTENÇÃO	3	0,32%
TOTAL	938,4	100,00%

TABELA 6-2 AGRUPAMENTO DAS ATIVIDADES NAS FASES.

Para cada fase é utilizada a expressão $\mathrm{PCf} = \frac{\mathrm{Hf}}{\mathrm{Ht}} \mathrm{x}\, 100$, onde PCf = percentual da fase sobre o total de horas, Hf = número de horas despendidas na fase e Ht = total de horas utilizadas no projeto.

6.2.3 DISTRIBUIÇÃO DAS PORCENTAGENS DOS PROGRAMAS POR FASE

A tabela de distribuição das porcentagens dos programas por fase (tabela 6-3) apresenta o percentual do tempo que deverá ser utilizado para cada um dos programas do projeto com base no número de pontos de função dos programas.



Módulo	PF	ANALISE	DESENVOLVIM	DOCUMENTAC	CQ\TESTES	IMPLANTACAC
Programa1	6,93	0,61	1,24	0,02	0,32	0,26
Programa2	16,83	1,48	3,01	0,04	0,77	0,62
Programa3	9,9	0,87	1,77	0,02	0,45	0,37
Programa4	22,77	2,00	4,08	0,06	1,05	0,84
Programa5	50,49	4,43	9,04	0,12	2,32	1,86
Programa7	6,93	0,61	1,24	0,02	0,32	0,26
Programa9	6,93	0,61	1,24	0,02	0,32	0,26
Relatórios	39,6	3,48	7,09	0,10	1,82	1,46
pr11	9,9	0,87	1,77	0,02	0,45	0,37
Tabelas	103,95	9,13	18,60	0,25	4,77	3,83
Programa6	3,96	0,35	0,71	0,01	0,18	0,15
Programa8	5,94	0,52	1,06	0,01	0,27	0,22
		24,96	50,85	0,69	13,04	10,50

TABELA 6-3 TABELA DE DISTRIBUIÇÃO DAS PORCENTAGENS DOS PROGRAMAS POR FASE.

Para fazer o calculo do percentual é utilizada a expressão:

$$PCp = \frac{PFp}{PFt} \text{ x } PCf$$

onde PCp = percentual do programa, PFp = pontos de função do programa, PFt = pontos de função total do projeto.

6.2.4 DISTRIBUIÇÃO DAS HORAS POR PROGRAMA/FASE

A tabela de distribuição das horas em programas/fase (tabela 6-4) apresenta o número total de esforço estimado para cada programa ao longo das fases do projeto.

Módulo	ANALISE	DESENVOLVIMENT	DOCUMENTACAO	CQ\TESTES	IMPLANTACAO
Programa1	5,71	11,63	0,16	2,98	2,40
Programa2	13,86	28,24	0,38	7,24	5,82
Programa3	8,15	16,61	0,23	4,26	3,42
Programa4	18,75	38,21	0,52	9,80	7,88
Programa5	41,58	84,73	1,15	21,73	17,46
Programa7	5,71	11,63	0,16	2,98	2,40
Programa9	5,71	11,63	0,16	2,98	2,40
Relatórios	32,61	66,46	0,90	17,04	13,70
pr11	8,15	16,61	0,23	4,26	3,42
Tabelas	85,60	174,45	2,37	44,74	35,95
Programa6	3,26	6,65	0,09	1,70	1,37
Programa8	4,89	9,97	0,14	2,56	2,05
	233,98	476,82	6,49	122,27	98,27

TABELA 6-4 TABELA DE DISTRIBUIÇÃO DAS HORAS EM PROGRAMAS/FASE.

Com esta estimativa é possível realizar a alocação de horas dos responsáveis de cada fase no projeto.

O número de horas dos programas por fase é dado pela expressão:

$$Hfp = PCp \text{ x } Ht$$

onde Hfp = número de horas por fase do programa.

6.2.5 DISTRIBUIÇÃO DE HORAS POR PROGRAMA

Por fim a tabela de distribuição de horas por programa (tabela 6-5) apresenta o total de horas estimadas para cada programa a ser desenvolvido.

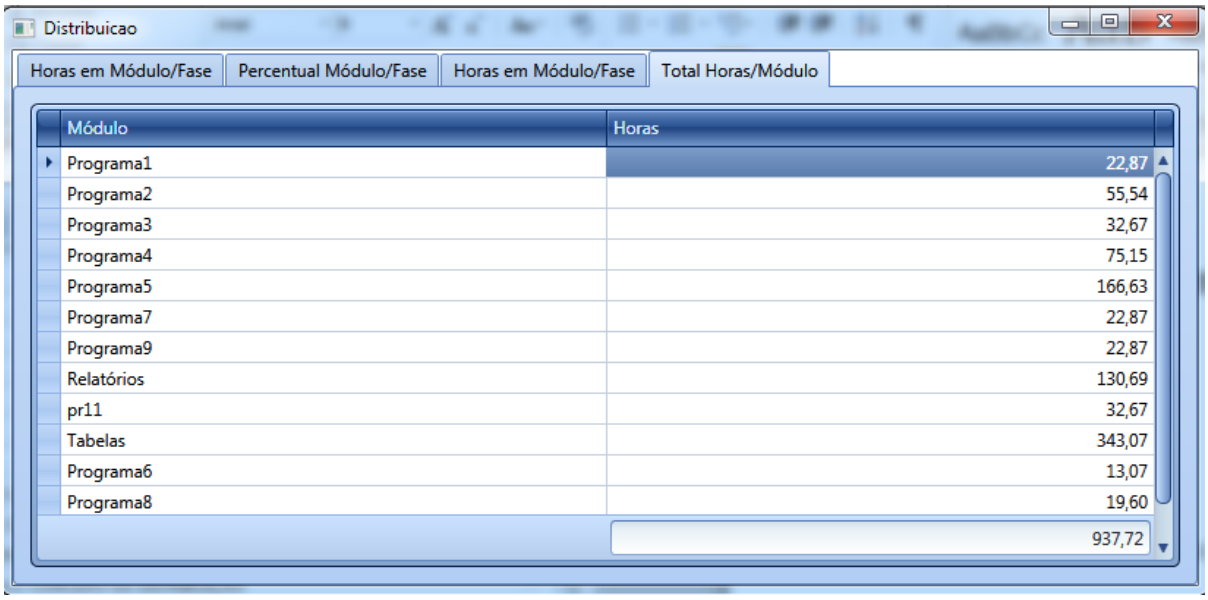


TABELA 6-5 TABELA DE DISTRIBUIÇÃO DE HORAS POR PROGRAMA.

Os valores são encontrados pela soma das horas de cada fase para cada programa como na expressa:

$$\mathrm{Hp} = \sum_{i=1}^{6} \mathrm{Hfpi}$$

onde Hp = horas por programa.

6.3 MODELAGEM DO SISTEMA

Nas próximas seções será apresentada a modelagem da ferramenta de distribuição de esforço pelas fases do projeto. De forma geral, para a programação da ferramenta, serão utilizadas as seguintes linguagens:

- C# (.NET): Linguagem orientada a objetos que será utilizada para definição e implementação das classes para a parte de regras de negócio do sistema;
- WPF (Windows Presentation Foundation): subsistema gráfico do framework .NET 3.0. Será utilizado na camada de apresentação para a definição e implementação dos objetos gráficos;

- Banco de Dados Access: Utilizado para armazenagem dos dados inseridos no sistema.

6.3.1 MODELO DE CASO DE USO

6.3.1.1 Caso de uso: Consultar estimativas do projeto

Nome: Consultar estimativa de horas do projeto

Descrição: Este caso de uso tem como finalidade consultar a estimativa de horas utilizando a produtividade e o número de pontos de função dos programas de um dado projeto.

Fluxo de Eventos:

Fluxo básico: Realizar estimativa de horas

- O sistema recupera as fases e percentual cadastrados;
- O Sistema recupera os programas e pontos de função cadastrados;
- O sistema calcula e exibe os resultados
- O sistema exibe quatro tabelas (Horas por fase, percentual por fase do programa, horas por fase do programa e total de horas do programa).
- O caso de uso é encerrado.

Pré condições:

- Não se aplica.

Pós condições:

- Não se aplica.

Regras de negócio:

- As regras de negócio foram apresentadas na seção do protótipo.

Fluxos alternativos:

- Não se aplica.

6.3.1.2 Caso de uso: Cadastro do Projeto

Nome: Cadastro de um projeto

Descrição: Este caso de uso tem como finalidade fazer o cadastro do projeto que será estimado. O usuário deverá fornecer a produtividade, os pontos de função de cada módulo e por fim adicionar as fases que deseja que sejam consideradas no projeto.

Fluxo de Eventos:

Fluxo básico: cadastrar o projeto

- O sistema recupera a média de produtividade dos projetos encerrados, se não houver qualquer projeto o usuário deverá fornecer o valor;
- O usuário adiciona os valores para os tipos de função de cada módulo do projeto;
- O usuário adiciona as fases e o percentual que deverão ser utilizados;
- O sistema calcula e exibe a quantidade de horas estimada para o projeto e o número total de pontos de função calculados;

Pré condições:

- Não se aplica.

Pós condições:

- Não se aplica.

Regras de negócio:

- As regras de negócio foram apresentadas na seção do protótipo e cálculo de pontos de função.

Fluxos alternativos:

- Não se aplica;

6.3.1.3 Caso de uso: Encerrar o Projeto

Nome: Encerramento de um projeto

Descrição: Este caso de uso tem como finalidade fazer encerramento de um projeto cadastrado no sistema. O usuário deverá fornecer o total de horas gastas em cada fase do projeto. Com essas informações o sistema pode calcular a produtividade realizada neste projeto, bem como o percentual utilizado em cada uma das fases.

Fluxo de Eventos:

Fluxo básico: encerrar projeto

- O sistema recupera as informações do projeto cadastrado;
- O sistema exibe as informações estimadas;
- O usuário insere a quantidade de horas realizada para cada fase do projeto;
- O usuário finaliza o projeto;
- O sistema calcula a produtividade realizada e armazena o valor;
- O sistema armazena o percentual de horas realizado para cada fase do projeto;

Pré condições:

- Obtenção das horas utilizadas para cada fase do projeto.

Pós condições:

- Não se aplica.

Regras de negócio:

- As regras de negócio foram apresentadas na seção do protótipo e cálculo de pontos de função.

Fluxos alternativos:

- Não se aplica;

6.3.2 MODELO DE ARQUITETURA

O modelo de arquitetura é de 3 camadas como representa a figura 6-1:

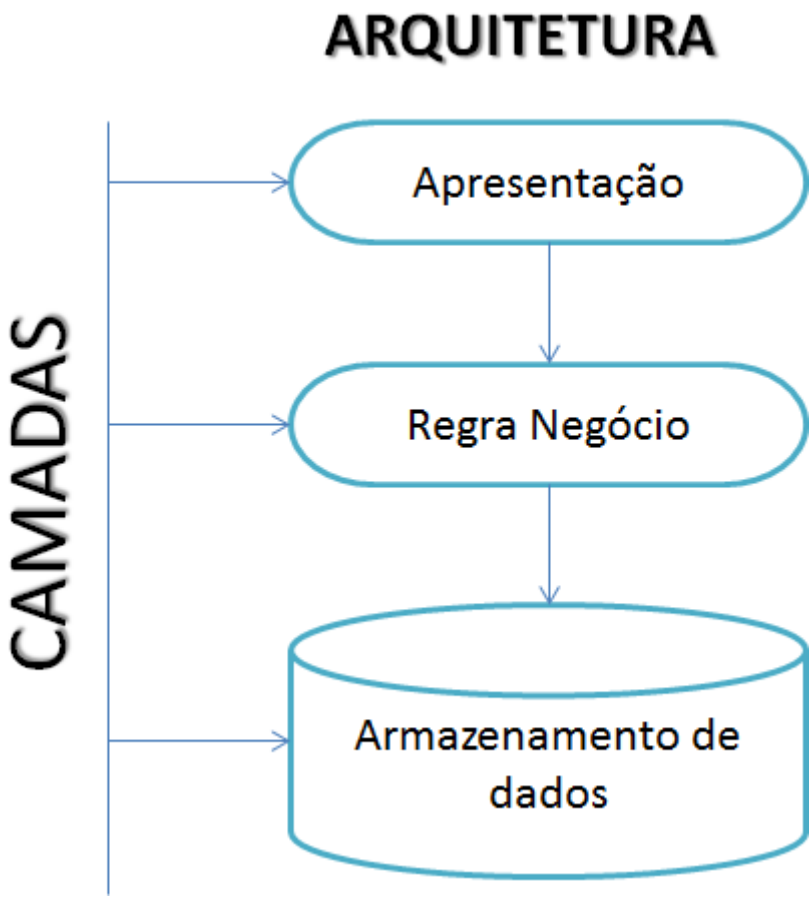


FIGURA 6-1 ARQUITETURA EM 3 CAMADAS.

A camada de apresentação contém todos os objetos gráficos do sistema, todo desenvolvimento realizado utilizando o WPF.

A segunda camada é contém as regras de negócio, onde se encontram todas as classes, métodos e fórmulas utilizadas no sistema. Esta camada é independente das outras, de forma que para se construir uma versão web deste sistema, bastaria implementar uma nova camada de apresentação, utilizando outra tecnologia, como por exemplo o ASP.NET.

Por fim, a camada de armazenagem de dados é composto por um banco de dados em Access, que é modelado a partir das próprias classes do sistema. Quem realiza esse processo é o próprio framework, ou seja, utilizando esta tecnologia não existe a necessidade de realizar uma modelagem de bancos de dados.

6.3.3 DIAGRAMA DE CLASSES

A figura 6-2 apresenta do diagrama de classes da camada de negócios. Para cada projeto cadastrado no sistema, haverá um ou mais módulos, uma característica geral do sistema e por fim, uma ou mais fases de desenvolvimento. Para cada módulo cadastrado existe um conjunto de tipos de função (Entrada Externa, Consulta Externa..) que são fatores essenciais para o cálculo do número de pontos de função de cada módulo.

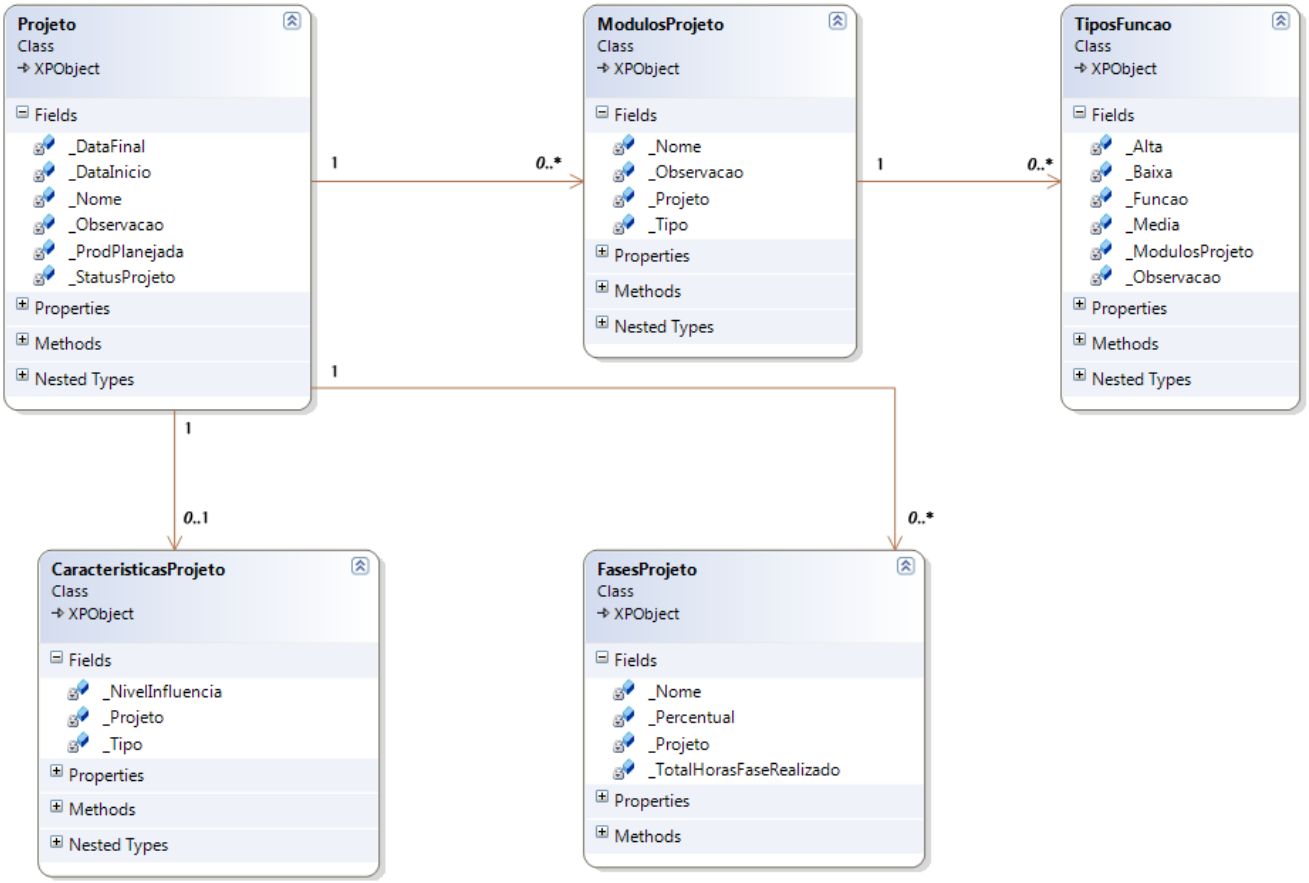


FIGURA 6-2 DIAGRAMA DE CLASSES PARA REGRA DE NEGÓCIOS

A figura 6-3 apresenta o diagrama de classes para a camada de apresentação. Esta camada consiste de uma tela inicial que realiza a chamada para a tela “ProjetosPage” que pode ser utilizada tanto para o cadastro de um novo projeto quanto a alteração do mesmo. Ainda nesta tela existe a chamada para a janela de adicionar os módulos (“AddModulo”) e definir as características gerais do sistema (“AddCGS”).

A tela inicial também possui uma chamada para uma tela de resumo do projeto. A partir desta tela é possível encerrar os projetos, gerar gráficos

gerenciais e também acessar a tela de distribuição de esforço pelas fases do projeto.

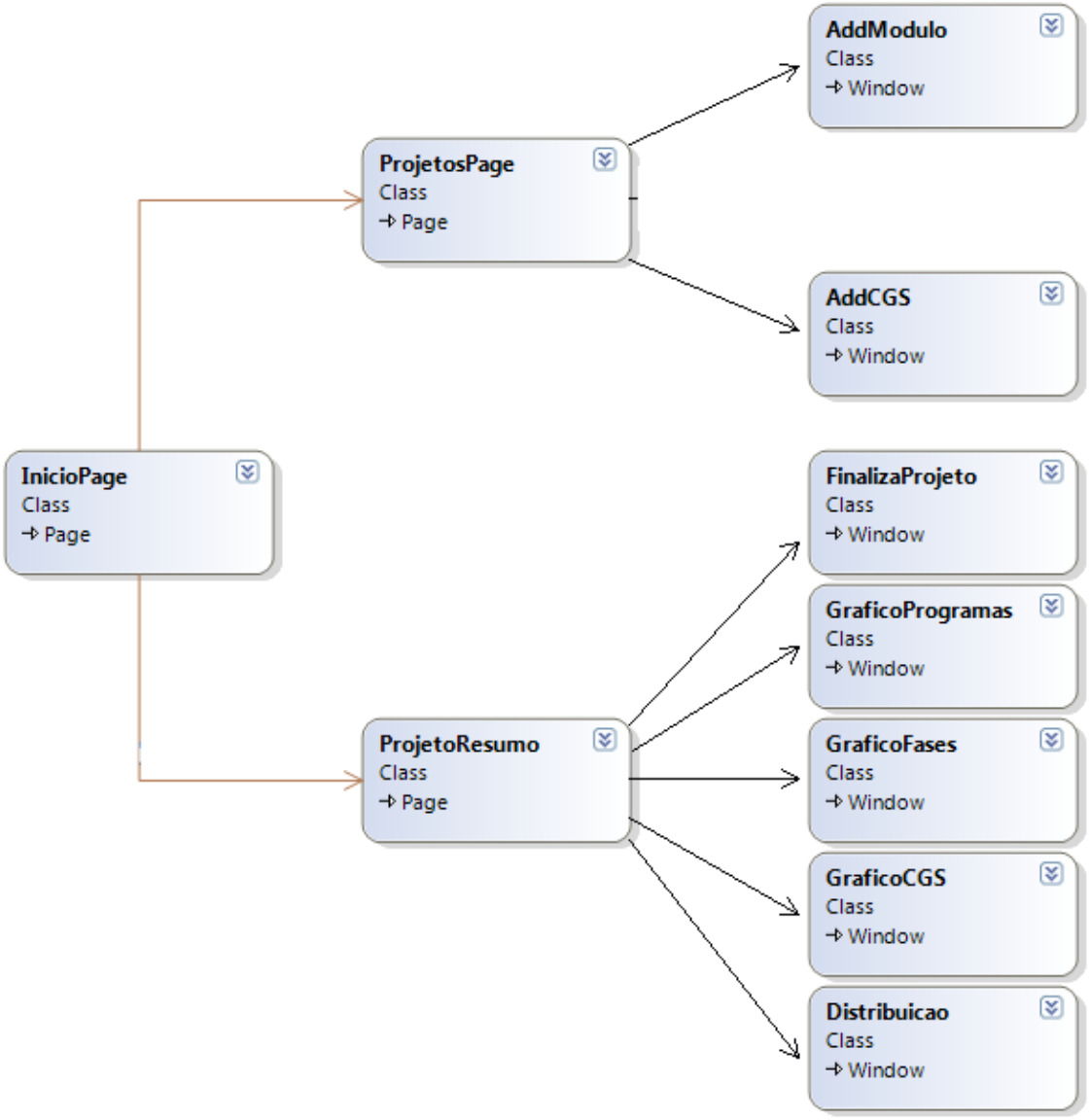


FIGURA 6-3 DIAGRAMA DE CLASSES DA CAMADA DE APRESENTAÇÃO

6.3.4 BANCO DE DADOS

A figura 6-4 apresenta o modelo de banco de dados. Ela representa basicamente a estrutura do diagrama de classes, seguindo a ideia de um projeto conter as fases e programas(módulos) do sistema.

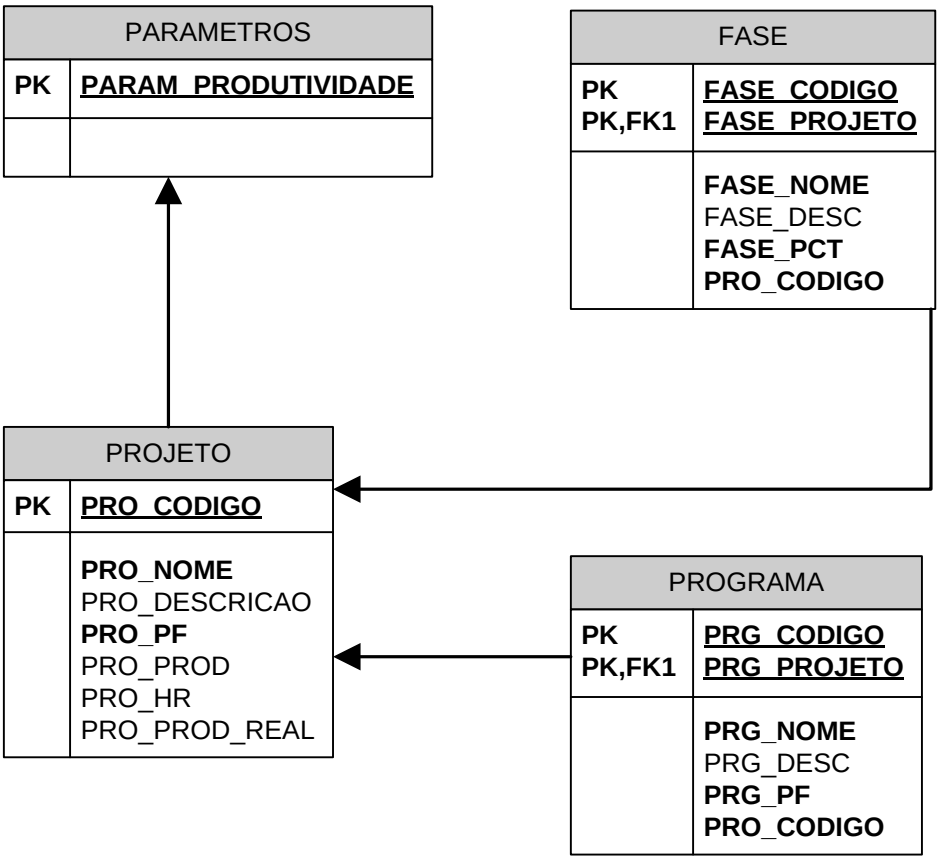


FIGURA 6-4 MODELO DE BANCO DE DADOS.

CONSIDERAÇÕES PARCIAIS

A ferramenta de distribuição pode ser muito útil para a empresa tanto na estimativa total de horas do projeto como a alocação de recursos pelas fases e programas do projeto. Esta fornece informações detalhadas do processo com basicamente a entrada de um parâmetro, o número de pontos de função dos programas que serão desenvolvidos.

Vale ressaltar que para a utilização da ferramenta será necessário o treinamento dos analistas de projeto para a contagem de pontos de função.

Além da distribuição de esforço, a ferramenta ainda pode ser utilizada como base histórica das produtividades de cada projeto e comparativo do que a ferramenta estimou com o realizado, possibilitando medidas para a melhoria.

7 RESULTADOS

Para realizar uma análise dos resultados proporcionados pela utilização da metodologia de medição e distribuição das horas, foi utilizado um novo projeto (ANEXO C), que por sua vez, também já havia sido concluído.

A tabela 7-1 apresenta os resultados obtidos para o novo projeto calculado.

	Horas	Produtividade (PF/H)	Produtividade (H/PF)
Planejado	289	0,473	2,115
Realizado	404,4	0,338	2,960
Diferença (P/R)	28,54%		

TABELA 7-1 CÁLCULO DA PRODUTIVIDADE DO NOVO PROJETO.

Novamente foi identificado um erro na estimativa de horas para o desenvolvimento do projeto. Embora a diferença percentual de horas tenha caído em relação ao projeto anterior, ela ainda persiste e numa quantidade elevada de 28% de horas utilizadas a mais do que o planejado.

Ainda na tabela 7-1, percebe-se que a produtividade aumentou para esse projeto, o que pode indicar um nível maior de maturidade dos participantes ou não foram encontradas tantas dificuldades no desenvolvimento do projeto em relação ao anterior.

Utilizando o próprio recurso da empresa, essa diferença de planejado e realizado já era visível, o importante dessa análise é obter um segundo parâmetro para a produtividade da empresa, onde pode-se fazer um refinamento para estimativas posteriores.

Agora, vamos considerar o cenário de forma que tivesse sido utilizada a produtividade encontrada no primeiro projeto (ANEXO A) para estimar as horas no novo projeto (ANEXO C).

Totais	PF
PF não ajustados	138
Fator de Ajuste	0,99
Pontos Função	**136,62**

TABELA 7-2 TOTAL DE PONTOS DE FUNÇÃO DO NOVO PROJETO.

Considerando que se obteve uma produtividade de 0,282 PF/H ou 3,550 H/PF e um total de 136,62 pontos de função, temos um total estimado de 485 horas para este novo projeto. A diferença percentual de horas planejadas e realizadas é de -19,93%, ou seja, nesta estimativa “sobraram” horas. O motivo desta diferença está na produtividade atingida pela equipe de desenvolvimento para este novo projeto (Tabela 7-2).

	Horas	Produtividade (PF/H)	Produtividade (H/PF)
Planejado	485	0,282	3,550
Realizado	404,4	0,338	2,960
Diferença (P/R)	-19,93%		

TABELA 7-3 CÁLCULO UTILIZANDO A PRODUTIVIDADE ENCONTRADA.

A partir desses dados seria fácil dizer que nenhum dos dois métodos (utilizado pela empresa x utilizado neste trabalho) funciona de forma ideal. Porém existe uma diferença muito grande entre estimar horas inferiores ao realizado a horas superiores ao realizado.

Neste momento, ao invés de existir um prejuízo com a estimativa, tem-se certa folga na quantidade de horas, ou seja, não haverá necessidade de realocar recursos pela não conclusão do trabalho no tempo planejado.

A tabela 7-3 mostra a diferença percentual de horas do que foi planejado e o realizado utilizando os dois métodos de estimativa.

Projeto 2	Estimativa por experiência	Estimativa por produtividade / APF
Planejado	289,00	485,00
Realizado	404,40	404,40
Diferença (P/R)	28,54%	-19,93%

TABELA 7-4 DIFERENÇA PERCENTUAL DE HORAS

Utilizando a metodologia a empresa terá uma maior aproximação do total de horas utilizadas para a realização do projeto, em quase 10% mais eficiente.

Por fim, vamos considerar a calibragem automática que o sistema realiza. Neste caso se dá pela média das produtividades gravadas no sistema.

Sempre que um projeto é finalizado, o usuário da ferramenta deverá inserir os dados realizados no programa para que em cima das horas reais gastas, seja efetuado o cálculo da produtividade para este projeto. Este valor aliado à produtividade do sistema resulta na nova produtividade, que será utilizada como base para novos projetos.

O cálculo é realizado a partir da formula a seguir:

$$Pn = \frac{Ps + Pp}{2}$$

onde Pn é a nova produtividade, Ps é a produtividade do sistema e Pp é a produtividade do projeto finalizado.

O objetivo desse cálculo é refinar e acompanhar o desenvolvimento da empresa, hora com aumento da produtividade ou com a queda da mesma, ou seja a ferramenta de estimativa deve acompanhar a evolução para fornecer resultados com maior precisão.

Considerando a média entre a produtividade encontrada em cada projeto, chegamos aos valores apresentados na tabela 7-4.

	Horas	Produtividade (PF/H)	Produtividade (H/PF)
Planejado	447,7	0,305	3,277
Realizado	404,4	0,338	2,960
Diferença (P/R)	-10,71%		

TABELA 7-5 CÁLCULO UTILIZANDO A PRODUTIVIDADE MÉDIA.

Fazendo o uso da produtividade média, chega-se a uma diferença de 10% a mais do que o realizado.

	horas	(%)
ANÁLISE	68,71	16,98%
DESENVOLVIMENTO	258,56	63,91%
DOCUMENTAÇÃO	7,2	1,78%
TESTES \ CQ	54,44	13,46%
IMPLANTAÇÃO	6,33	1,56%
MANUTENÇÃO	9,16	2,26%
	404,4	

TABELA 7-6 PERCENTUAL DE HORAS POR FASE DO PROJETO

A tabela 7-5 apresenta a distribuição de horas pelas fases do projeto, que em comparação ao projeto anterior são bastante semelhantes. Com a utilização do sistema, esse percentual também será calibrado/atualizado a cada projeto realizado.

CONSIDERAÇÕES PARCIAIS

O objetivo desse capítulo foi apresentar os benefícios alcançados com a utilização da metodologia, embora não alcance a situação ideal, a aproximação entre a previsão e a realidade dos projetos é evidente.

Para que a situação ideal seja alcançada é fundamental que as métricas e os indicadores obtidos sejam revisados durante todo novo ciclo de desenvolvimento, aprimorando seus valores através da base histórica dos resultados obtidos anteriormente.

CONCLUSÃO

Os objetivos estabelecidos por este trabalho foram alcançados. Com relação à identificação do processo de desenvolvimento na empresa, a medição do tamanho funcional de um projeto da empresa para obtenção da produtividade e por fim a criação de uma ferramenta que dado o número de pontos de função calculados do projeto apresenta, além do total de horas estimadas, toda a distribuição de esforço pelas fases definidas do projeto.

O método de contagem por pontos de função de fato, parece ser o mais adequado para medições de projeto na empresa, visto que apresentou números condizentes. O fato de o método independer da linguagem é primordial para o sucesso da contagem, visto que na empresa são utilizadas 3 camadas de diferentes linguagens, no caso a apresentação, regras de negócio e banco de dados. O conceito de tamanho funcional da um bom panorama de todo o projeto, permitindo realizar o calculo da produtividade próxima a real.

Levando em consideração a evolução do conhecimento daqueles que desenvolvem o programa, foi considerado que a produtividade não seria fixa, portanto a utilização da média de todos os projetos será sempre adotada para o desenvolvimento posterior, de modo que, a produtividade seja refinada a cada projeto.

As grandes dificuldades do projeto estão no próprio cálculo dos pontos de função, que necessitam de muita atenção aos detalhes do projeto para que o tamanho funcional esteja de acordo com o que será realizado e assim obtendo uma estimativa mais precisa.

Por fim, é importante ressaltar que esta ferramenta será de grande valia para empresa. Visto que não existe a utilização de uma metodologia na estimativa de horas, além de associar as atividades a um grupo de fases e ter a visão da distribuição das horas ao longo destas, também é aplicado um novo conceito, para empresa, de análise do projeto, utilizando os pontos de função.

TRABALHOS FUTUROS

Existem alguns aspectos interessantes que podem ser adicionados no futuro nessa ferramenta, abaixo seguem algumas ideias:

- Integração dos dados como banco da empresa para obtenção dos dados, bem como atualização em tempo real do esforço utilizado, representando com cores se está abaixo, dentro de uma margem ou acima do esforço estimado;
- Criar um ambiente de cálculo de pontos de função, onde o usuário este para inserir os dados do cálculo dos pontos de função que automaticamente serão atribuídos aos programas do módulo de distribuição de esforço.
- Apresentar gráficos da evolução da produtividade, horas por fase etc, para facilitar a análise do gerente de projetos para que tome as medidas de correção.

REFERÊNCIAS BIBLIOGRÁFICAS

BELL Canada Inc.; **Trillium: Model for Telecom Product Development and Suport Process Capability**, release 3.0, Dec. 1994.

BOAS, André Luis de C. V. **Qualidade e Avaliação de Software**. UFLA. 2003.

BRABARÁN, Gabriela; FRANCISCHINI, Paulino. **Indicadores de Produtividade na Indústria de Software**. USP, 2002.

DAVENPORT, Thomas H. **Reengenharia de Processos: Como inovar na empresa através da tecnologia da informação**. 5 ed. Rio de Janeiro: Campus, 1994.

FALBO, **Ricardo A. Engenharia de Software**. UFES – Universidade Federal do Espírito Santo. 2005.

FENTON, N. **Software metrics: A Rigorous and Pratical Approach**. PWS Publishing Company, 1991.

FRANCISCHINI, Paulino; BRABARÁN, Gabriela. **Critérios Considerados pelas Indústrias de Software para Avaliação dos seus Projetos**. USP, 2000.

ISO 9000-3, **Normas de Gestão da Qualidade e Garantia da Qualidade - Diretrizes para aplicação da NBR 19001 ao desenvolvimento, Fornecimento e Manutenção de Software**, Rio de Janeiro, 1993.

KEZNER, Harold. **Projecto Management – A Systems Approach to Planning, Scheduling and Controlling**. 1998.

PAULK, Mark et al. **The Capability Maturity Model – Guidelines for Improving the Software Process**, CMU-SEI, Addison-Wesley, 1994.

PECK, Chris; Callahan, Dale. **A Proposal for Measuring Software Productivity in a Working Environment**. 2006.

PRESSMAN, Roger S. **Software engineering : a beginner's guide.** 1988.

PRESSMAN, Roger S. **Software engineering : a practitioner's approach.** 2005.

PROSYST, Desenvolvimento de Sistemas. 2008. Acessado em 01 de outubro de 2008. Disponível em: http://www.prosyst.com.br.

SALVIANO, **Clênio Figueiredo. Melhoria e Avaliação de Processo com ISSO/IEC 15504 (SPICE) e CMMI**. 2004.

SAYÃO, Jefferson L. A. **A Automação dos Processos Como Forma de Integração da Pequena e Média Empresa ao Comércio Eletrônico e a Cadeia de Suprimentos**. Dissertação de mestrado. UFSC. 2004.

SEI, **Website do Software Engineering Institute** – SEI. Acessado em outubro de 2008, http://www.sei.cmu.edu/.

SHAW, M. **Prospects for an Engineering Discipline of Software**, IEEE Software, 1990.

SHEARD, A. Sarah. **The Framework Quagmire, A Brief Look**. Agosto, 1997.

SPINOLA, **Mauro de Mesquita. ISO 9000 para Software.** UFLA, 2004.

TICKIT Project; **Guide to Software Quality System Construction and Certification using EN29001**, 1992.

TRINDADE, André et al. **COCOMO II – uma compilação de informações sobre a nova métrica.** USP, 1999.

USC, **Website Center for Systems and Software Engineering – USC.** Acessado em outubro de 2008, http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html

VASCONCELOS, Alexandre. **Introdução à Engenharia de Software e aos Princípios de Qualidade**. UFLA/FAEPE, 2003.

VAZQUEZ, Carlos Eduardo. **Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de Software.** ÉRICA/SP, 2003.

ANEXO A

PROJETO DO SISTEMA DE RECEBIMENTO DO LEITE

ANEXO B

TABELA DE HORAS GASTAS NO PROJETO

ANEXO C

PROJETO DO CONTROLE DO REAJUSTE DE PREÇOS DE VENDA DOS PRODUTOS

ANEXO D

TABELA DE HORAS GASTAS NO PROJETO

ANEXO E

EXEMPLO DA MEDIÇÃO DOS PROCESSOS ELEMENTARES