UNIVERSIDADE DO EXTREMO SUL CATARINENSE

CURSO DE CIÊNCIA DA COMPUTAÇÃO

FRANCIELLE WARMLING NUERNBERG

ESTUDO COMPARATIVO DE MÉTRICAS DE SOFTWARE PARA UTILIZAÇÃO EM EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE DE PEQUENO PORTE

CRICIÚMA, DEZEMBRO DE 2007

FRANCIELLE WARMLING NUERNBERG

ESTUDO COMPARATIVO DE MÉTRICAS DE SOFTWARE PARA UTILIZAÇÃO EM EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE DE PEQUENO PORTE

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientadora: Profa. MSc. Ana Cláudia Garcia Barbosa

CRICIÚMA, DEZEMBRO DE 2007

Dedico este trabalho aos meus amigos e familiares por todo o apoio recebido ao longo de minha formação e por todo seu amor, confiança e compreensão.

AGRADECIMENTOS

Agradeço a todas as pessoas do meu convívio que acreditaram e contribuíram, mesmo que indiretamente, para a conclusão deste curso.

Aos meus pais João e Ladi que se esforçaram nos bons e maus momentos desses anos para que eu alcançasse os meus objetivos. Ao Cletson por estar sempre ao meu lado me ouvindo e apoiando minhas decisões. A querida tia Leonidas com sua bondosa ajuda me levou adiante nos períodos difíceis. Aos irmãos e meus cunhados e especialmente a pequena Sarah que chegou à família para dar mais alegria e encher nossos corações de paz com seu rosto de anjo. Aos meus grandes amigos e colegas que me incentivaram a continuar adiante. À minha orientadora Ana Cláudia Garcia Barbosa em especial pela compreensão, paciência e empenho por ter acreditado na possibilidade da realização deste trabalho.

Meus sinceros agradecimentos às empresas de desenvolvimento de software da região carbonífera por se disponibilizarem a responder a pesquisa e contribuir para a realização do trabalho. Agradecimento em especial: Bauhaus Sistemas Ltda, Consulti Tecnologia da Informação, Consystec Consultorias e Sistemas Ltda, Ema Software, Lorenzi Informática, PD Sistemas, Sixnet – Systems Integration Experts, Top Informática, TWCOM Sistemas, Useall Software Ltda e 4S Assessoria e Desenvolvimento de Sistemas Ltda.

“Há várias medidas para medir a vontade humana.
A mais exata e a mais segura é a que se exprime por
esta questão: de que esforço és capaz?”
William James

RESUMO

Esta pesquisa realizou estudo e comparação dos métodos de métricas de software com os requisitos das empresas para estimar custos e prazos mais precisos e ter melhor visão do progresso do projeto.

Para as empresas de software, um dos aspectos atuais que constitui as maiores preocupações na área de engenharia de software é estimar e cumprir as metas de prazo e custo dos projetos. Pois as estimativas no final do projeto podem estar subestimados ou superestimados. A causa principal é a falta de credibilidade na predição dos dados por essas empresas se basearem apenas na experiência dos gerentes de projetos e não em utilizar um método de auxílio. Como uma tentativa de minimizar este problema, torna-se necessário um método de métrica de software para auxiliar o gerente. Entretanto é necessário que este método esteja adequado ao perfil da empresa.

Palavras-Chave: Engenharia de Software; Gerenciamento de software; Métricas de software.

ABSTRACT

This research conducted a study and comparison between the methods of software metrics and the companies request, in order to estimate costs and more accurate terms and have a better vision of the project progress.

For software businesses, one of the current aspects which is part of the greatest concerns in the area of software engineering, is to estimate and meet the deadlines and cost of the projects, for the estimates at the end of the project may be underestimated or overestimated. The main cause of that is the lack of credibility in predict the data, for these companies are based only on the experience of the project managers and not in making use of an auxiliary method. As an attempt to minimize this problem, there is a need of a method of software metrics to help the manager. However, it is necessary that this method meets the company profile.

Keywords: Software Engineering; software Management; software metrics.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

AIE	Arquivos de Interface Externa
ALI	Arquivos Lógicos Internos
CE	Consulta Externa
COCOMO	*Constructive Cost Model*
CMM	*Capability Maturity Model*
CMMI	*Capability Maturity Model Integration*
EE	Entradas Externas
EF	*Environment Factor*
FA	Fator de Ajuste
PFA	Function Point Analysis
PFNA	Ponto de Função Não Ajustados
ISO/IEC	*International Organization of Standardization / International Electrotechnical Commission*
LOC	*Lines Of Codes*
PMBOK	*Project Management Body of Knowledge*
PMI	*Project Management Institute*
SE	Saídas Externas
SW-CMM	*Capability Maturity Model for Software*
TFC	*Technical Complexity Factor*
UAW	*Unajusted Actor Weights*
UCP	*Use Case Point*
UUCP	*Unadjusted Use Case Points*
UUCW	*Unajusted Use Case Weights*

SUMÁRIO

1 INTRODUÇÃO

As transformações constantes no mercado de desenvolvimento de software e a competitividade organizacional transformam as estruturas, os processos e as estratégias das empresas (MOLINARI, 2004). Para se adaptar e sobreviver nesse mercado competitivo o aprimoramento de técnicas de apoio ao desenvolvimento de software torna-se necessário, para assim as organizações evoluírem na melhoria no desempenho dos softwares.

Nos projetos de softwares, as estimativas e mensurações dos custos, tempo e esforços exigidos, dependem diretamente do planejamento feito pelo gerente de desenvolvimento. Quanto mais precisa melhora a qualidade e a produtividade da empresa.

A Engenharia de Software oferece como método de auxílio as métricas de software, capazes de gerar dados para os gerentes analisarem e utilizarem como melhoria no projeto e no processo de desenvolvimento de software.

Atualmente existem vários métodos de métricas de software sendo utilizadas como apoio pelos gerentes. Nesse trabalho os métodos abordados são os seguintes: Linhas de Código Fonte (LOC); Análise de Ponto de Função (FPA); Ponto de Caso de Uso (UCP) e Modelo de Custo Construtivo (COCOMO).

O gerente de desenvolvimento que não utiliza um método de métricas de software, conta apenas com sua experiência profissional e por mais experiente que seja, ele se depara com projetos que ainda não trabalhou. Com isso há uma dificuldade de estimar o prazo de implantação, custo, qualidade e produtividade da equipe, o que pode ocasionar uma perda de confiabilidade com o cliente. Portanto, uma empresa de

desenvolvimento que não utilizar um método de auxílio à estimativa pode apresentar os seguintes pontos problemáticos (GRANDCHAMP, 2002):

a) após análise final do software, o custo está acima do preço estimado no início do projeto;
b) menos conhecimento dos recursos necessários para o desenvolvimento;
c) sem indicativo de produtividade da equipe de projeto e desenvolvimento;
d) menos precisão na ordem cronológica dos recursos;
e) cronograma com mais alterações ao longo do projeto;
f) dificuldade na identificação de riscos no desenvolvimento e estratégias de administração dos mesmos;
g) sem monitoramento e controle das tarefas e dos prazos estipulados;
h) perda de qualidade do software.

Porém, a utilização de um método de métricas de software por si só não garante que estes problemas não ocorram, pois não basta utilizá-la, é necessário que ela esteja adequada ao estilo de desenvolvimento da empresa.

Dessa forma, esta pesquisa propõe um estudo detalhado dos métodos de estimativas, para verificação dos que melhor satisfazem as empresas de desenvolvimento de software de pequeno porte.

1.1 OBJETIVO GERAL

Realizar um estudo comparativo dos métodos de métricas de software, para utilização em empresas de desenvolvimento de pequeno porte.

1.2 OBJETIVOS ESPECÍFICOS

De modo a permitir a obtenção do objetivo geral estabelecido, os seguintes objetivos específicos foram alcançados:

a) compreender métricas de software;
b) analisar como as empresas de desenvolvimento de software procedem no planejamento de projetos;
c) comparar os métodos de métricas de software;
d) analisar a compatibilidade dos métodos de estimativas com os requisitos das empresas de desenvolvimento de software de pequeno porte;

1.3 JUSTIFICATIVA

O planejamento de um software é de fundamental importância para o desenvolvimento de um projeto. Para esse projeto ser planejado corretamente torna-se necessário um método de métrica para auxiliar o gerente de desenvolvimento a fazer as medições mais precisas.

Realizar uma pesquisa nas empresas que desenvolvem software comercialmente permite conhecer se utilizam um método de métricas de software ou baseia-se na experiência dos gerentes de projetos. Quais as prioridades que as empresas teriam para um método de métricas de software adequada a sua funcionalidade.

Com as necessidades identificadas, realizar-se-á uma comparação dos métodos de estimativas mais conhecidas como: Linhas de Código Fonte (LOC); Análise de Ponto de Função (FPA); Ponto de Caso de Uso (UCP) e Modelo de Custo Construtivo (COCOMO), para verificação dos métodos de estimativa de softwares que mais se adequam as empresas de desenvolvimento de software de pequeno porte.

O gerente que utiliza um método de métrica de software poderia ter um acompanhamento mais eficiente do desenvolvimento de um projeto. Como também maior precisão do custo do projeto de software e uma avaliação da produtividade das pessoas que desenvolvem o software. O gerente também terá maior conhecimento dos recursos necessários para o projeto e uma melhor precisão na ordem cronológica desses recursos, menos alterações no cronograma estipulado e conhecimento dos esforços exigidos.

1.4 ESTRUTURA DO TRABALHO

A estrutura do trabalho está organizada em 6 capítulos. Este capítulo 1 contextualiza a introdução referente ao trabalho, com a definição do objetivo desse estudo e também os objetivos específicos e concluindo então com a justificativa do trabalho realizado. O gerenciamento de projetos abordado no capítulo 2 visa apresentar desde o seu surgimento nas organizações, o papel do gerente frente a um projeto até os elementos importantes para um planejamento de sucesso. Dentre esses elementos está às métricas de software, citado no capítulo 3, fornecendo aos gerentes métodos de auxílio para estimar os seus projetos. Esses métodos de métricas são apresentados no capítulo 4 e dentre e os mais conhecidos na engenharia de software e citado no trabalho são os métodos LOC, FPA, UCP e COCOMO.

O capítulo 5 mostra os trabalhos que possuem correlação com este estudo.

Por fim é abordado no capítulo 6 a aplicação da pesquisa nas empresas de desenvolvimento de software de pequeno porte, no qual foi analisado como era o gerenciamento nessas organizações e ao final a compatibilidade dos métodos com os requisitos da empresa.

2 GERÊNCIA DE PROJETOS

O cenário atual das empresas de desenvolvimento de software se caracteriza pelas constantes mudanças no mercado de desenvolvimento, pelas novas alternativas de tecnologia disponíveis e principalmente pela extrema competitividade organizacional (PERALTA; BULLA, 2000).

Devido a esses fatores, para as empresas continuarem competitivas num mercado que está cada vez mais exigente, devem estar em constante desenvolvimento. Principalmente em aprimorar seus serviços como, por exemplo, buscar melhor qualidade no software e projetos com estimativas mais precisas. Pois com isso obtém vantagens consideráveis sobre suas concorrentes por ser um fator de importância num projeto de sucesso.

O importante para os clientes é apresentar um software de qualidade que venha a resolver seus problemas nos negócios. E para isto, as empresas de desenvolvimento de software buscam no gerenciamento de projeto atender o cliente. Pois gerenciamento de projetos é aplicar os conhecimentos, habilidades e técnicas nas atividades do projeto que visam alcançar o objetivo proposto pelo cliente (PMI, 2000).

A gerência de software engloba desde o objetivo do software a ser implementado, os riscos que afetarão o andamento, os recursos exigidos para desenvolvimento, o controle das atividades durante a produção e o custo do software até o final da implantação do software. Ou seja, o gerenciamento abrange todo o desenvolvimento do software, não se atem apenas no planejamento inicial do projeto (PRESSMAN, 1995).

2.1 HISTÓRIA DO GERENCIAMENTO DE PROJETOS

O gerenciamento de projetos de software surgiu no final da década cinqüenta nos Estados Unidos e desde então vem evoluindo e modificando com o passar do tempo. Porém foi somente formalizada no início da década de sessenta, quando perceberam que a forma do processo dos projetos de desenvolvimento de software eram precários. Nesse tempo os resultados nem sempre eram atendidos e o gerenciamento em uma organização era muito mais conflitante. Além do mais esse processo em grandes projetos era muito burocrático e com uma geração enorme de papel para formalizar qualquer ação. E, além disso, acrescenta Prado (2004), as ocorrências de algumas experiências fracassadas inibiam a introdução do gerenciamento em outros ramos de negócios e deixou, além disso, uma má fama, que retardou uma maior expansão do gerenciamento nas indústrias de informática.

Os aspectos considerados nessa década até os meados de setenta na gerência de Projetos de Software era o Planejamento, Custo e Qualidade (PCQ). Mas foi a partir daí que o escopo passou a ser visto como essencial no processo. Esse período foi conhecido como Gerenciamento de Projetos Tradicional. Sua abordagem era centrada em aspectos técnicos e aspectos como Recursos Humanos e atendimento ao cliente não eram as suas prioridades (MOLINARI, 2004).

O fim da época do gerenciamento tradicional ocorreu quando perceberam que a satisfação do cliente deve ser o principal item de sucesso de um projeto e que os projetos são realizados por uma equipe de pessoas que dependem da sua produtividade para o bom andamento do projeto.

O Gerenciamento de Projetos Moderno surgiu com uma abordagem ampla na década de 90. Marcado com o surgimento de um padrão universal de gerenciamento

de projetos o *Project Management Body of Knowledge* (PMBOK) publicado pelo *Project Management Institute* (PMI) dos Estados Unidos, estabelece uma constituição que se difundiu como uma disciplina de conhecimento básico que os gerentes de projetos devem conhecer. E continua sendo bastante utilizado e incrementado até hoje.

O gerenciamento de projetos no Brasil começou a ser praticado na década de sessenta. Entre as décadas de sessenta e setenta grandes organizações adotaram e propagaram essa técnica em seus setores. As organizações nessa época tinham pouca concorrência, o que não acelerava a motivação para aperfeiçoamento das técnicas de gerências. Mas com a globalização pelo ano de 1992 as empresas rapidamente se dirigiam a novos padrões de qualidade e produtividade.

Atualmente por existir uma grande competição entre as organizações e as estratégias mundiais apontam no sentido de inovação, as empresas brasileiras estão dando maior importância ao gerenciamento de projetos para a sua sobrevivência e progresso.

"O Brasil está se destacando na seguinte área: melhoria dos procedimentos de rotinas, focando em qualidade, redução de gastos e aumento de receita. Esforços desta natureza permitirão as empresas uma melhor competitividade internacional" (PRADO, 2004, p. 37).

A evolução do gerenciamento de projetos sucede com as transformações que estão ocorrendo no mundo. São fatores como a competitividade, maior cobrança e exigência de precisão fazem a gerência de projetos assumir um papel contínuo em transformação.

2.2 O GERENTE DE DESENVOLVIMENTO DE PROJETOS DE SOFTWARE

Segundo Aldabó (2001, p. 18) “o gerente de desenvolvimento é responsável pela obtenção dos objetivos do projeto e pela liderança da equipe”. E para que isso aconteça cabe ao gerente planejar, estimar e mensurar o custo, o tempo e esforços exigidos de um projeto para realizarem as suas metas.

O gerente de desenvolvimento desempenha uma função de desafio, explicado por sua grande responsabilidade, por habilidades esperadas ou desejadas num gerente e a liderança para manter um bom relacionamento com a equipe e conduzi-los ao objetivo do projeto. Pois qualquer deslize pode comprometer seriamente o projeto tornando um fracasso total ou perda em desempenho no andamento do projeto.

É difícil definir o trabalho de um gerente, pois varia muito dependendo da organização que ele vai atuar ou o projeto de software a ser desenvolvido. Segundo Valeriano (1998) os gerentes têm que estar aptos a desempenhar as seguintes metas e ações principais para realizar o planejamento dos projetos:

a) estabelecer o objetivo do projeto;
b) organizar a equipe de projetos;
c) definir recursos, processos e tecnologias necessárias e levantar fontes;
d) estabelecer um cronograma;
e) estimar custos e prazos;
f) coordenar o planejamento detalhado do projeto;
g) estabelecer os mecanismos de controle;
h) fazer estimativas e mensuração do projeto;
i) monitorar a evolução do projeto;
j) tomar decisões caso identifique um conflito em seu andamento;

k) neutralizar antecipadamente os itens de riscos;

l) controlar a qualidade do software, entre outras metas e ações.

Portanto, o gerente de projeto de software torna-se fundamental para a sobrevivência e o crescimento da empresa no mercado de desenvolvimento de software. Pois uma gerência errada pode acarretar em muitas falhas no projeto. Como exemplos pode-se citar:

a) as estimativas são feitas em base empírica por não utilizar uma ferramenta automatizada de estimativa ou um mal uso da ferramenta;

b) prazos estipulados abaixo do tempo real de desenvolvimento;

c) projeto sem acompanhamento dos riscos e dos progressos do avanço físico do projeto.

Esse e outros motivos que podem ocasionar atrasos, falhas e perdas de qualidade num projeto de desenvolvimento de software. Assim fica ressaltada e evidenciada a importância do gerenciamento para o sucesso do projeto.

2.3 ELEMENTOS-CHAVES DA GERÊNCIA DE DESENVOLVIMENTO DE PROJETOS

Pressman (1995) cita que no gerenciamento de software existem elementos-chaves para que os gerentes de projetos se concentrem e principalmente analisem com prioridade esses elementos para desempenhar um projeto de software bem sucedido. Os elementos-chaves que serão descritos ao longo do trabalho são:

a) métricas e qualidade de software;

b) estimativas;

c) análise dos riscos;

d) determinação de prazos;

e) monitoração e controle.

No gerenciamento de software esses elementos-chaves seguem diferentes funções no projeto. As métricas fornecem informações quantitativas para auxiliar na tomada de decisão e a qualidade garante um produto mais confiável ao cliente. O objetivo das estimativas é avaliar os dados do projeto para apresentar os índices ao pessoal envolvido no projeto. Realizar uma análise dos riscos é prever as mudanças que poderão ocorrer ao longo do projeto. Com a determinação dos prazos definimos as atividades que serão realizadas e por fim a monitoração e o controle permitem assegurar que o objetivo do produto seja realizado e também o acompanhamento das alterações que ocorrerem no projeto, fazendo o devido ajustamento com o projeto.

2.3.1 Métricas e Qualidade de Software

O desenvolvimento de software tem em sua composição um importante processo para garantir que o cliente da empresa receba um produto dentro das normas por ele definida e esperada. Esse processo de desenvolvimento caracteriza pelas definições e especificações relevantes de qualidade do produto, com a mensuração que fornece uma avaliação de adequação e progresso evolutivo do projeto e do software.

Na seção 3.1 serão apresentados alguns conceitos relacionados à qualidade de software e em seguida alguns padrões mais utilizados nos processos de desenvolvimento. A seção 3.2 é dedicada a métricas de software, que tem por objetivo auxiliar as organizações em tomadas de decisões do projeto.

2.3.2 Estimativas

A cada projeto de desenvolvimento de software os gerentes se deparam com um software ou um projeto diferente, mas que utiliza algumas tarefas, tamanhos e funções iguais aos projetos anteriores. Então cabe ao gerente de projetos prever as mudanças que ocorrerá em cada software, para estimar com maior precisão os esforços, prazos e custos que implicará esse novo projeto.

Porém a estimativa de um projeto pode se tornar complexo se o software apresentar características inovadoras em relação aos anteriores (VAVASSORI, 2002). Estimar um projeto pode necessitar de experiência, acesso a informações históricas e utilizar técnicas de estimativas (métricas de software) como base para que dê segurança e que forneça confiança para os projetos a que venham suportar. Além disso, o gerente que documentar os dados auxiliará e facilitará posteriormente um melhor acompanhamento, como também fazer novas estimativas no andamento do progresso do projeto, quando esse ocorrer um desvio das atividades, ajudará a manter o projeto no objetivo estipulado.

De acordo com Cantor (1998 apud BORREGO FILHO, 2003) as estimativas geralmente são realizadas em cinco áreas: prazos, custos, tamanho, esforço e produtividade e para a projeção de cada uma destas estimativas, existem várias técnicas e métodos, como pode-se citar as métricas de software que serão descritas na seção 3.3.

As estimativas de tamanho são consideradas a base principal para determinar o prazo, esforços e custos adequados. Destes, os esforços e os custos são gerados por dados históricos de projeto concluídos. Já o prazo inicialmente é gerado a partir da experiência dos gerentes, considerando o tamanho e esforços do projeto e

tamanho da equipe de desenvolvimento. As estimativas de produtividade geralmente se baseiam em medir alguns atributos do software e dividir esse resultado pelo esforço total (SOMMERVILLE, 2003).

2.3.3 Análise de riscos

Quando um gerente planeja um projeto é esperado que esse software seja desenvolvido com qualidade, seja confiável e que os prazos e custos estejam previamente determinados no projeto. No entanto, em qualquer projeto há incertezas em determinar e planejá-los corretamente. Ou seja, o gerente de projetos de software pode não prever os riscos que afetarão o projeto.

O risco de um projeto consiste na probabilidade de que um evento ou uma incerteza ocorra no desenvolvimento do projeto de software (MOLINARI, 2004). Por esse motivo a análise dos riscos eficaz tem um papel tão importante na obtenção de sucesso de um projeto de software.

Os riscos decorridos podem afetar várias áreas. Na área de projeto os riscos afetam a programação ou os recursos do projeto. Na área de software os riscos agem na qualidade ou no desempenho do software que está em desenvolvimento. Finalmente na área dos negócios afetam a organização que está desenvolvendo como também quem está adquirindo o software.

No ano de 2002, de acordo com *Standish Group* (2004), foi desenvolvido um estudo nas empresas de desenvolvimento de software dos Estados Unidos, denominado "*Chaos*", com o objetivo de averiguar os destinos de seus projetos de software. Na pesquisa constatou-se que 66% dos projetos de software apresentavam erros de previsão de cronograma, orçamento e/ou qualidade; destes 66% apenas 15%

dos projetos foram cancelados e 51% dos projetos foram concluídos, porém acima do orçamento e em atraso, ou ainda incompleto em relação às características e funções especificadas originalmente. Em 2004, realizou-se outra pesquisa, na qual apresentou um aumento no índice de falhas de projetos de software e alterações do escopo inicial do cronograma de 2 %.

Percebe-se que na pesquisa apresentada a existência de um aumento no índice de falhas de projetos e aqueles concluídos, no entanto com alterações no cronograma e nas estimativas de custo. A partir desta constatação, surge a necessidade da identificação dos fatores de risco de um projeto de software, pois para as empresas de pequeno porte, se os riscos não forem bem identificados e analisados podem determinar o futuro da organização.

Após um levantamento e análise dos riscos do projeto os resultados devem ser implantados no projeto juntamente com as ações corretivas, ocorrências de riscos e conseqüências dos mesmos na organização ou no desenvolvimento do software e ao final monitorar e controlar constantemente os possíveis riscos que surgirão ao longo do projeto para as devidas ações corretivas.

O importante é que o gerente de projetos deve prever os riscos e que compreenda o impacto do mesmo no projeto ou na organização, para que assim tome as devidas providências.

2.3.4 Determinação de Prazos

Outro aspecto importante na gerência de projetos, ressaltado por Pressman (1995) é a determinação de prazos no desenvolvimento de software.

Um projeto consiste em sua composição uma série de atividades a serem realizadas ao longo do seu desenvolvimento. Algumas dessas diferentes dos projetos anteriores e outras tarefas se relacionam com qualquer projeto de engenharia.

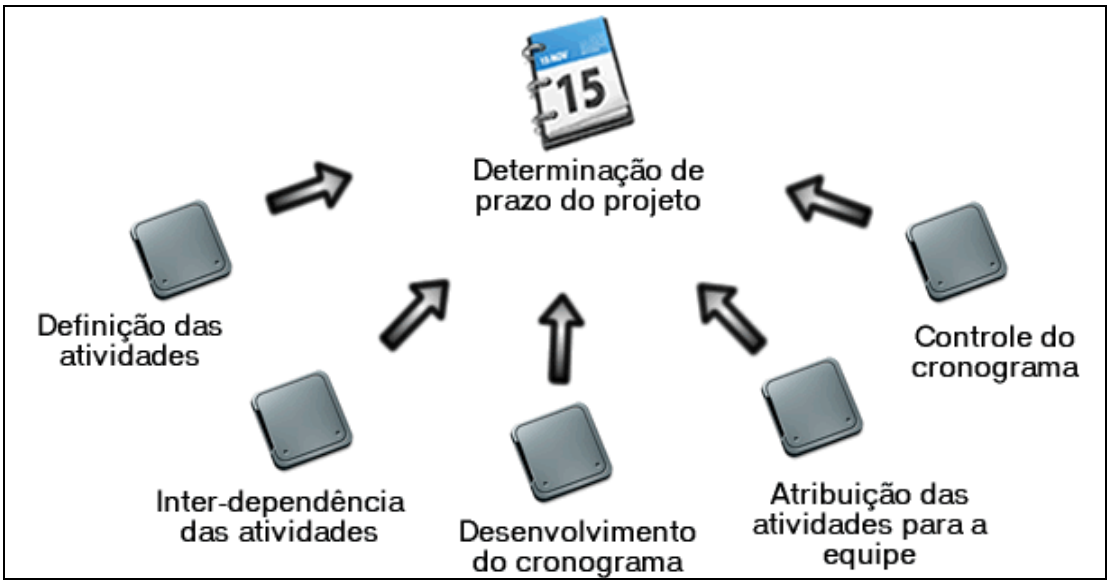

Figura 1. Determinação de Prazo
Fonte: BORREGO FILHO, L. (2003).

A definição de um cronograma do projeto, como demonstrou a Figura 1, consiste no geral de cada atividade identificada em atribuir período de realização, os responsáveis e os inter-relacionamentos com as outras atividades, identificar a atividade e os esforços a serem atribuídos a cada membro da equipe de projeto, a definição do prazo de realização e sua data final de entrega.

> A precisão na determinação de um cronograma às vezes pode ser bem mais importante do que a precisão na determinação dos custos. Num ambiente orientado para o produto, os custos adicionais podem ser absorvidos por nova determinação de preço ou por amortização ao longo de um grande número de vendas. Um cronograma descumprido, porém pode reduzir o impacto de mercado, criar clientes insatisfeitos e elevar os custos internos (PRESSMAN, 1995, p. 141).

Portanto é fundamental o acompanhamento das atividades do projeto descritos no cronograma e que o mesmo esteja refletindo a situação atual do projeto por meio de atualizações e alterações periódicas das informações nele contidas.

2.3.5 Monitoração e Controle

Controlar um projeto consiste em acompanhar o progresso das atividades em execução, com a finalidade de assegurar a realização de objetivos estabelecidos. Uma monitoração efetuada regularmente permite a interpretação das informações, com o intuito de avaliar e ajustar o desempenho com as metas estabelecidas no planejamento (ALDABÓ, 2001).

Na medida em que são identificadas as falhas no desenvolvimento do software, o gerente identifica o impacto que essa falha venha ocorrer na execução do projeto. O gerente pode até utilizar uma ferramenta de monitoração e controle automatizado para ajustar esse desvio no andamento do desenvolvimento do projeto aos prazos e custos estabelecidos inicialmente.

O controle é necessário em todos os sistemas, por atuar mais como ação preventiva em vez de corretiva, pois age sobre as causas e os processos não atacando conseqüências ou retrabalhos nas falhas (GOETS, 2003).

As mudanças no projeto são inevitáveis e necessárias. Algumas estarão sobre o controle do gerente de projetos se este monitorar e acompanhar as metas estabelecidas. Pois pode ser detectada a necessidade de mudar o planejamento para evitar uma falha potencial. Mas antes tem-se que avaliar o impacto no projeto e analisar sua ação nos prazos e custos antes de alterar o planejamento.

O gerente de projetos deve ter como objetivo principal antecipar dificuldades e crises, evitando soluções de urgência e improvisadas, perdendo com isso qualidade e credibilidade do projeto e do cliente.

3 MÉTRICAS E QUALIDADE DE SOFTWARE

Em um software, além de assegurar que será entregue no prazo estimado e dentro do custo previsto é ideal que atinja a qualidade estabelecida.

Para isso acontecer é importante definir ainda durante o planejamento do projeto os padrões de qualidade a serem seguidos e as medições a serem realizadas. Assim, contudo, realizar melhorias na qualidade do produto e estimativas mais precisas.

3.2 QUALIDADE

Qualidade não se define apenas em ausência de defeitos ou algo que atende às especificações, mas em olhar para um produto com confiabilidade, usabilidade, utilidade e capacidade de manutenção e entre outros (YOURDON, 1995).

As empresas de desenvolvimento atualmente se empenham em desenvolver produtos de software que tem por objetivo atingir um alto nível de qualidade. Pois pela busca de mercado entre as correntes da área, produzir um software de baixa qualidade conduz a erros no andamento do projeto como também problemas e reparos futuros após entrega do produto ao cliente.

A qualificação do software pode ser avaliada sobre dois pontos de vista: A primeira seria o fator explícito, no qual a avaliação é realizada pelas expectativas do cliente; A segunda avaliação seria um fator implícito, que é levado em consideração os fatores percebidos pelos desenvolvedores. Vários autores idealizam que a qualidade do software pode ser qualificada pela intensidade a qual satisfaz as necessidades do cliente e também pelo atendimento a requisitos técnicos (BEZERRA, 2001).

Por esse motivo pode-se afirmar que a qualidade de software é a sua capacidade em cumprir certas especificações, seja do produto ou da organização. As especificações não seão perfeitas, mas é preciso algum esforço para melhorá-las. Ao gerente de desenvolvimento cabe a responsabilidade de garantir que a qualidade do produto atinja o nível exigido (SOMMERVILLE, 2001).

Uma forma de buscar e comprovar a qualidade no processo de desenvolvimento de software são as certificações *International Organization of Standardization/International Electrotechnical Commission* (ISO/IEC) ou *Capability Maturity Model* (CMM), assim, desta maneira, se tornar competitivas em um mercado cada vez mais exigente. A Tabela 1 mostra os padrões ISO de qualidade de produtos de software mais conhecidas e utilizadas no desenvolvimento de software.

Tabela 1. Padrões de qualidade de software ISO/IEC

Tipos de ISO		***Descrição***
ISO 8420		Define qualidade em termos da habilidade de satisfazer necessidades dos usuários explícitas e implícitas.
ISO/IEC 9126	9126-1	Preocupa-se em conceituar e avaliar a qualidade do produto de software sob o ponto de vista do usuário, dentro de um ambiente e um contexto estabelecido de utilização.
	9126-4	Apresenta um conjunto de métricas categorizadas por características baseadas na norma de qualidade em uso de software.
ISO/IEC 12207		Descreve processos para desenvolvimento e manutenção de software, fornecendo um conjunto essencial de atividades que deverão ser completadas para obter um produto de software.
ISO/IEC 14598	14598-1	Apresenta a estrutura de funcionamento da série de normas para avaliação da qualidade dos produtos de software.
	14598-2	Contém requisitos e guias para atender funções de avaliação dos produtos
	14598-3	Destina-se ao uso durante o processo de desenvolvimento e manutenção
	14598-4	Estabelece um processo sistemático para avaliação dos softwares pré-desenvolvidos, visando a aceitação dos mesmos.
	14598-5	Fornece orientações para a implementação prática da avaliação do produto de software, quando necessitam aceitar os resultados da avaliação.
	14598-6	Explica como desenvolver módulos de avaliação e como validá-los.
ISO/IEC 15504		Propõe a avaliar projetos e organizações de forma a verificar o nível em que estas se encontram, em termos de maturidade dos processos utilizados e das práticas organizacionais no desenvolvimento de projetos de software.

Fonte: WEBER, K; ROCHA, A. (1999), BEZERRA, C. (2001).

A ISO realiza uma avaliação de como a empresa de desenvolvimento faz o projeto de software. Por outro lado, o CMM ajuda as empresas de desenvolvimento a escolher uma estratégia gradual e evolutiva para melhorar o processo desde a sua ausência de controle até o mais otimizado.

O CMM teve início em 1988, sob coordenação de Watts S. Humphrey do *Software Engineering Institute* (SEI) dos Estados Unidos. Surgiu com o objetivo de desenvolver um produto integrado de suporte ao processo e a melhoria de produto. Inicialmente baseava-se em um questionário de maturidade que ajudava a determinar as prioridades nas organizações de melhoria nos seus processos. Em 1991, o SEI evolui a estrutura para o chamado *Capability Maturity Model for Software* (SW-CMM), que visa melhorar o processo para engenharia de software. O SW-CMM define cinco níveis de maturidade e as áreas chaves de processos (exceto o primeiro nível). Os níveis baseiam-se numa escala que ao atingirem suas metas, em termos de custo, prazo, funcionalidade e produtividade, oferecem uma melhora contínua nos processo de software (BORREGO FILHO, 2003).

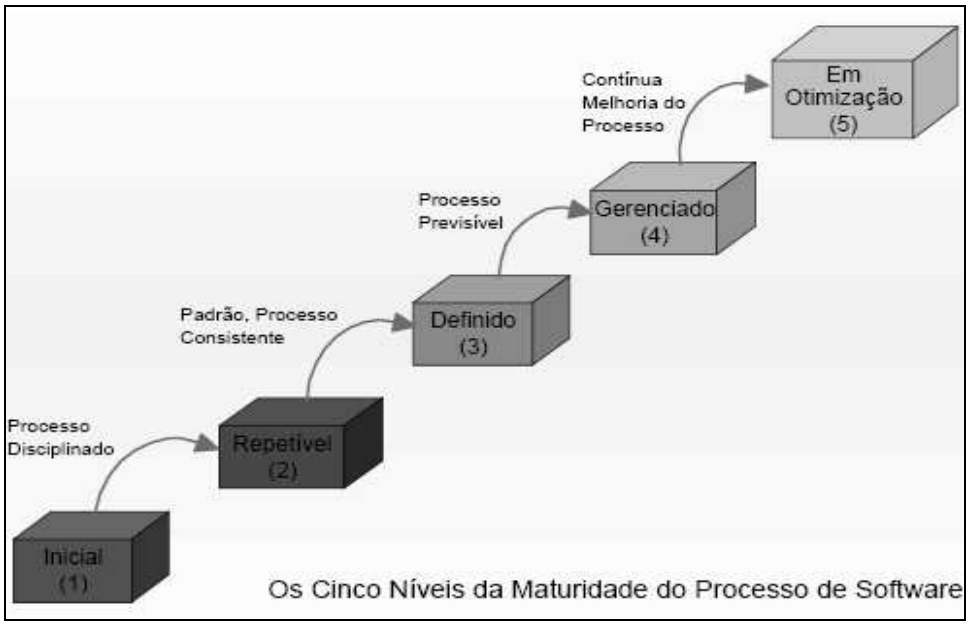

Figura 2. Níveis CMM
Fonte: Machado, A. (2004).

A Figura 2 demonstrou os cinco níveis do CMM, as características de cada nível descritos a seguir, apresentam mais detalhada a distinção que cada nível se aplica:

a) inicial (Nível 1): o processo de desenvolvimento não é organizado e não há um controle sobre o andamento e os possíveis riscos do projeto. O sucesso, se ocorrer, depende de esforço individual;

b) repetível (Nível 2): o planejamento e o controle são definidos como processos básicos de gerência. Os processos estabelecidos servem para repetir o sucesso dos projetos anteriores em outros projetos que venham a ser semelhantes.

c) definido (Nível 3): todos os processos de software, desde a área de gerência até a engenharia, estão documentados, padronizados e integrados. os projetos utilizam um modelo padrão de processo de desenvolvimento de software para toda a empresa;

d) gerenciado (Nível 4): o processo de desenvolvimento e os produtos são medidos detalhadamente, para serem compreendidos e controlados quantitativamente, ou seja, o foco é a qualidade do produto para melhor planejamento a partir das medições realizadas. A organização estabelece as métricas de forma a medir as características do produto;

e) otimizado (Nível 5): refere-se ao melhoramento contínuo do processo através de realimentação dos mesmos, indicadores dos controles e pelo uso de idéias e tecnologias inovadoras.

Uma certificação CMM evidencia que a organização está em um determinado nível de maturidade, mas para estar em uma maturidade, os níveis anteriores também devem ser atendidos.

Esta estrutura do CMM foi projetada para dar suporte as empresas em aplicações como, por exemplo, identificar os pontos fortes e as melhorias a serem colocadas em prática na organização. Como também entender as atividades necessárias para realizar o planejamento e a implementação de um software. E por fim organizar o tempo de desenvolvimento evitando retrabalho (BORREGO FILHO, 2003).

Com a evolução do modelo SW-CMM, em 2000 foi lançado um novo modelo, chamado *Capability Maturity Model Integration* (CMMI). O CMMI tem como objetivo realizar a integração da representação em estágios contínuos. As representação em estágios seria semelhante a abordagem SW-CMM, mas com algumas alterações. No qual se baseia em utilizar um conjunto de áreas de processos para mostrar a empresa como alcançar a maturidade. Já a representação contínua, demonstra que a organização faz a escolha de áreas individualmente (MARRECO, 2006).

Para as pequenas empresas a representação contínua seria mais adaptável, pois evoluiria com a empresa e também facilidade de adoção por pequenas equipes.

O Brasil se encontra em 14ª lugar dentre os países com maior número de avaliações CMM realizadas pelo instituto SEI. Até agosto de 2006 o índice de certificados CMM no Brasil era de 49 organizações. Destas, 40 contam com certificado de nível 2, 8 organizações tem certificado de nível 3 e apenas uma possui certificado de nível 4 (COUTO, 2007).

Seja qual a forma de padronização e certificação adotada, a empresa estará mostrando aos clientes a credibilidade em oferecer um produto de qualidade. Além de para a própria empresa apresentar os pontos de melhoria que necessita para manter-se cada vez a frente do mercado.

3.3 MÉTRICAS DE SOFTWARE

As métricas de software têm como objetivo obter a mensuração dos indicadores quantitativos dos atributos de um produto ou de um processo de software.

Na engenharia de software as métricas vêm se tornando um dos seus principais focos. Pois as empresas de desenvolvimento buscam na qualidade, rapidez e nos custos reduzidos o diferencial para obter vantagens consideráveis sobre as concorrentes e manter a credibilidades com os clientes que adquirirem um sistema na empresa. Contudo, os gerentes de projetos ainda não têm o hábito de utilizar métricas de software como apoio para geração de dados a serem analisados no projeto.

Os principais motivos são citados por muitos autores: uma razão seria que apesar do progresso de técnicas e ferramentas de medições, as abordagens ainda são ultrapassadas ou inadequadas para as tecnologias de sistemas que a empresa utiliza. Outro motivo seria o desinteresse das empresas em adaptar essas técnicas na empresa. E por fim a abordagem métricas de software ainda é de baixa popularidade entre os profissionais da área.

A medição pode ajudar a determinar a evolução do projeto, melhorar o processo de desenvolvimento do produto e apresentar de forma clara os dados para que todas as pessoas envolvidas no projeto entendam o processo do software (VAVASSORI, 2002). Além de outros benefícios que a métricas de software oferecem, de acordo com Pressman (1995) e Matos Júnior (2006) eles seriam:

a) maior controle do processo de desenvolvimento;
b) possibilidade de criar estimativas a mais precisas;
c) redução de custos e tempo de desenvolvimento;

d) aumento da satisfação e confiança do cliente, devido a maior qualidade do software;

e) melhor produtividade da equipe de desenvolvimento da organização.

Apesar de mostrar os benefícios que a utilização de métricas nos proporciona, existe ainda uma dificuldade em sobre o que medir e que interpretações nos oferecem. Há dificuldade principalmente em decidir qual é o método mais apropriado ou até questionar a validade de comparações envolvendo pessoas, produto e processos.

À empresa de desenvolvimento cabe a responsabilidade de buscar o conhecimento do objetivo de métricas e avaliar qual método proporcionará os benefícios para melhorar continuamente o processo de desenvolvimento de software.

As métricas de software têm várias classificações, mas a principal origina em qual aspecto a medição deseja atingir:

a) métricas de produto: faz medições relativas somente ao software final ou o produto intermediário;

b) métricas de processo: fornece os dados sobre o procedimento de desenvolvimento como um todo.

As medições podem ser classificadas em outra classe, conforme Pressman (1995) e citados por Matos Júnior (2006):

a) métricas diretas: contêm dados que não dependem de outras medições. As métricas diretas de processo podem citar os custos e esforços aplicados. Já as métricas diretas de produto podem citar o número de linhas produzidas;

b) métricas indiretas: as características analisadas dependem das medições de uma ou mais características. As medições indiretas do produto podem-

se citar como exemplo a funcionalidade, qualidade, complexidade, eficiência e muitas outras.

Pressman (1995) ainda cita que o domínio de métricas pode-se dividir em mais categorias. A primeira categoria seria a métricas de produtividade que se baseia na saída do processo de desenvolvimento. A segunda seria a métrica de qualidade que faz uma indicação do software conforme as exigências do cliente. A terceira categoria são as métricas técnicas que se concentram na característica do software. Sobre uma nova visão podemos ter uma segunda divisão, na qual seriam a métricas orientadas ao tamanho que visa medições diretas da saída e da qualidade. As métricas orientadas a função que realiza medições indiretas do software e do processo. E por fim as métricas orientadas as pessoas que informam sobre a forma como as pessoas desenvolvem o software.

Existem atualmente no mercado de desenvolvimento de software vários métodos de métricas para realizar medições de estimativas. Alguns desses métodos bastante utilizados e outros ainda não são de tanto conhecimento nas empresas. Dentre os mais conhecidos, que serão descritos com mais detalhes na seção 4, encontra-se:

a) Linhas de Código-Fonte (LOC);
b) Análise de Ponto de Função (APF);
c) Pontos por Caso de Uso (UCP);
d) Modelo de Custo Construtivo (COCOMO).

Os métodos de métricas de software têm como objetivo fornecer dados quantitativos, por exemplo, estimativa de tamanho, esforços e prazos. Cada um desses métodos possui uma diferente técnica de medição.

4 MÉTODOS DE MÉTRICAS DE SOFTWARE

Para se chegar a uma medida de software existem vários tipos de métodos de software que avaliam as variáveis de tamanho, esforço e prazo. O método LOC é baseado na contagem de linha de código e tem por objetivo verificar o tamanho final do produto. Já o APF fornece dados segundo a visão e os requisitos do usuário final. Com o surgimento do desenvolvimento orientado a objetos o método UCP fornece dados estimativos com base na contagem de caso de uso e o COCOMO apresenta as estimativas necessárias para o projeto. Desses métodos o LOC é considerado um dos mais simples de ser realizado.

4.2 LINHAS DE CÓDIGO FONTE

Classificado como métrica orientada ao tamanho, o LOC (*Lines Of Codes*) foi um dos primeiros métodos de estimativas de software. Surgiu na década de cinqüenta e mede o tamanho físico de linhas de código do programa fonte. A ideologia do método LOC é que se um sistema possui um número LOC maior que outro então esse sistema será mais complexo.

Embora essa métrica pareça simples, existem vários fatores que podem influenciar no resultado de sua contagem. O estilo do programador pode ser um fator na contagem das linhas de código fonte e principalmente a linguagem de programação utilizada. Como por exemplo, pode-se citar a programação em assembler, em que cada linha de código corresponde a um comando. Nessa linguagem a definição é clara na métrica, mas utilizando uma linguagem de alto nível a métrica pode ser influenciada,

dentre outras coisas, pela consideração ou não de comentários, linhas em brancos e declarações que não são executadas (FIGUEIREDO, 2006).

A métrica de LOC apresenta impossibilidade de uso em estimativas, considerando que a contagem de linhas de códigos só poderá ser feito depois da codificação. Então não é de grande valia na etapa de planejamento do projeto de software. Mas como a métrica de LOC é de fácil obtenção e de simples aplicação a qualquer projeto de software, torna-se uma boa condição para quem inicia a utilizar métricas em seus projetos.

4.3 ANÁLISE DE PONTO DE FUNÇÃO

A Análise de Ponto de Função (FPA - *Function Point Analysis*) foi introduzida em 1979 por Allan J. Albrecht da IBM, sendo posteriormente refinado em uma metodologia formal e publicado no domínio público em 1984. Dois anos depois foi formado o Grupo Internacional de Usuários de Ponto de Função (IFPUG) que teve por objetivo efetuar padronizaçõe adicionais nas regras da FPA.

Em 2002 passou a condição de padrão internacional, por meio da norma ISO/IEC 20926. “Mas na norma não é realizado o Cálculo do FA, pois o ajuste proposto pela FPA é atualmente questionado devido à grande variação na interpretação e a constatação que alguns estão desatualizados para a maioria das plataformas atuais” (GUERRA, 2006).

Classificada como métricas orientadas a função, FPA tem como objetivo realizar estimativas baseadas na medição do valor das funções executadas pelos programadores. Este método tem como perspectiva a contagem das funções na visão do usuário e não do programador (VAVASSORI, 2002).

Sua aplicação pode ser realizada ao final das definições de requisitos do software. Isso se deve por se basear em dados que serão mais bem conhecidos e interpretados logo no começo do desenvolvimento do projeto. Diferente da métrica de LOC, a FPA é independente da linguagem de programação utilizada no sistema ou do estilo do programador (PRESSMAN, 1995).

Associado com outras medidas a FPA permite obter as medições de produtividade, qualidade e custo do desenvolvimento do software. Sendo um benefício, pois além de medir o tamanho do produto, acompanha melhor a produtividade da equipe, melhora o controle da qualidade e os custos de desenvolvimento e manutenção são estimados mais precisos. Podendo-se apontar ainda outros benefícios gerais na utilização de FPA, citados por Braga (1996 apud VAVASSORI, 2002) e Guerra (2006):

a) melhorar o gerenciamento do projeto;
b) medir a satisfação do usuário em relação às soluções desenvolvidas e o processo empregado;
c) prever melhor o projeto por possibilitar realizações de estimativas nas fases iniciais do desenvolvimento do software;
d) justificar a necessidade de recursos e de pessoal;
e) possibilitar reestimativas;
f) melhorar o processo de desenvolvimento e manutenção, através da análise de pontos fortes e fracos e custos das falhas.

Pressman (1995) referencia que a contagem da FPA se baseia em dados subjetivos e não objetivos e também que não possui nenhum dado físico direto, pois é só um valor numérico.

4.3.1 Etapas da Contagem

A análise de ponto de função apresenta sete etapas para a realização da contagem:

a) determinação do tipo de contagem;

b) fronteiras de aplicação;

c) classificar e contar as funções do tipo dados;

d) classificar e contar as funções do tipo transação;

e) determinar os pontos de função não ajustados;

f) calcular o fator de ajuste;

g) calcular os pontos de função ajustados.

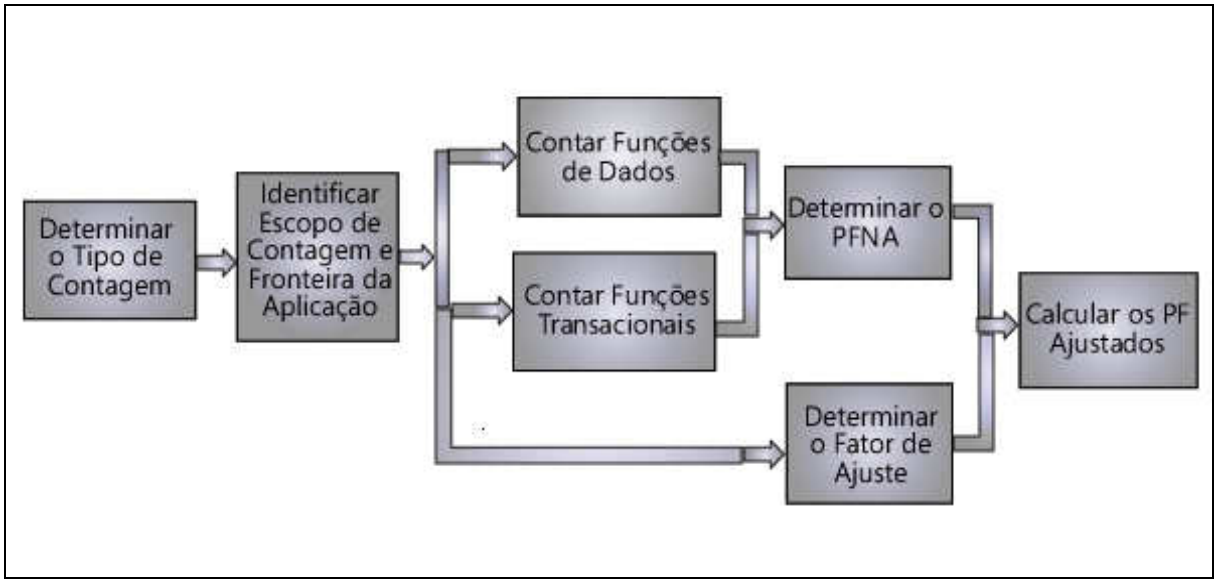

Figura 3. Procedimento do Ponto de Função
Fonte: HAZAN, C. (2000 apud VAVASSORI, F., 2002).

A ordem do procedimento das etapas foi mostrada na Figura 3. Os detalhes do que consiste cada uma das atividades está no APÊNDICE A.

4.4 PONTOS DE CASO DE USO

O Ponto de Caso de Uso (UCP – *Use Case Point*) foi introduzido em 1993 por Gustav Karner, com o objetivo de estimar projetos com base na contagem de casos de usos. Surgiu como uma adaptação do Ponto de Função para sistemas que são desenvolvidos em orientação a objetos (OO). A contagem da UCP pode variar devido à variação de estilo de casos de uso e a inexistência de padrões universais dificultando a comparação entre projetos de diferentes organizações.

A métrica PCU apresenta seis etapas de realização da contagem:

a) cálculo do Valor de Impacto dos Autores (UAW);

b) cálculo do Valor de Impacto dos Casos de Uso (UUCW);

c) determinar o Ponto de Caso de Uso Não Ajustado (UUCP);

d) cálculo do Fator de Ajuste da Complexidade Técnica (TCF);

e) cálculo do Fator de Ajuste da Complexidade de Ambiente (ECF);

f) determinação do Ponto de Caso de Uso Ajustado (UCP).

O modo como será realizado a contagem do Caso de Uso está no APÊNDICE B.

4.5 MODELO DE CUSTO CONSTRUTIVO

O modelo COCOMO (*Constructive Cost Model*) é caracterizado por realizar estimativas empíricas, ou seja, é realizada a estimativa considerando uma amostra de projetos anteriores. Este método busca medir o esforço necessário para desenvolvimento, o prazo e o custo total do software. Desde que tenha o tamanho do software através dos métodos LOC, FPA ou UCP (LÓPEZ, 2005).

Este método foi publicado no ano de 1981, no qual originou o nome COCOMO 81. Mas com os avanços tecnológicos, houve uma reformulação do modelo para adequar-se a essas novas tecnologias. Então em 1997 foi apresentado o novo modelo chamado de COCOMO II. Nas seções seguintes apresentam-se os detalhes de cada modelo.

4.5.1 Cocomo 81

O modelo COCOMO 81 foi desenvolvido pela *University of Southem Califórnia* em 1981, por estudo realizado pelo Professor Dr. Barry Bohem sobre 63 projetos de grande porte (GUERRA, 2006). O dado fundamental para o COCOMO 81 é tamanho do número de linhas de código (LOC). Sendo uma desvantagem, pois o LOC pode variar em função da linguagem de programação utilizada.

A estimativa realizada pelo COCOMO 81 é baseada em três modelos hierárquicos de Bohem:

a) básico: cálculo simples que computa o esforço, o custo como uma função do tamanho do software expresso em LOC estimados;
b) intermediário: computa o esforço como uma função do tamanho do programa e de um conjunto de "direcionadores de custo" que incluem avaliações subjetivas do produto, do hardware, do pessoal e dos atributos dos projetos;
c) avançado: incorpora o modelo intermediário com uma avaliação sobre o impacto dos direcionadores de custos sobre cada fase do projeto.

O COCOMO 81 pode ser aplicado em três tipos de projetos de software:

a) modo orgânico: para equipes relativamente pequenas;

b) modo semidestacado: quando a equipe é composta por técnicos experientes;

c) modo embutido: quando o projeto precisa operar em um ambiente complexo.

4.5.2 Cocomo II

Em 1997 foi lançado o COCOMO II devido a idade dos projetos que baseavam o modelo COCOMO 81. A base para esse modelo foi feita em cima de 161 projetos selecionados (GUERRA, 2006).

Como o modelo COCOMO 81 o COCOMO II também visa medir o esforço, prazo e custo desde que tenha como base o tamanho do software (LOC, FPA, UCP). O modelo ainda prevê um adicional de 20% ao tempo computado, como margem de erro (LÓPEZ, 2005).

O COCOMO II se baseia em um tipo de hierarquia, diferente do modelo anterior, que considera a fase que o projeto se encontra (LÓPEZ, 2005):

a) Composição da aplicação: Utiliza-se nas fases iniciais do projeto;

b) pré-projeto: utiliza-se quando a estrutura básica já estiver definida;

c) pós-arquitetura: aplica-se quando o software já está desenvolvido.

O valor para determinar os esforços, prazos e custos podem variar dependendo da hierarquia a ser utilizada, pois possui coeficientes e expoentes próprios que são ajustados de acordo com a fase do projeto. Isto é identificado como calibração do modelo (GUERRA, 2006).

Ao utilizar um método empírico deve-se ter em mente que os resultados obtidos podem não ser precisos, mas oferece um auxílio para analisar o projeto.

5 TRABALHOS CORRELATOS

Durante os estudos desse trabalho de pesquisa, foram analisados alguns trabalhos científicos com propósitos semelhantes a essa pesquisa, mas seus objetivos focados eram outros. Alguns trabalhos envolvendo métricas de software selecionadas estão descritos a seguir.

5.2 METODOLOGIA PARA O GERENCIAMENTO DISTRIBUÍDO DE PROJETOS E MÉTRICAS DE SOFTWARE

Tese de Fabiane Barreto Vavassori para obtenção do grau de Doutor em Engenharia de Produção, em 2002, pela Universidade Federal de Santa Catarina localizada em Florianópolis no estado de Santa Catarina.

A tese tem como objetivo apresentar uma metodologia suportada por uma ferramenta CASE de apoio aos gerentes. A qual emprega a tecnologia de agentes, para viabilizar o planejamento e o gerenciamento distribuído de projetos agregando métricas de software (VAVASSORI, 2002). E esse trabalho abordou uma comparação das ferramentas de métricas de software. O trabalho em questão pode ser encontrado no endereço: http://teses.eps.ufsc.br/defesa/pdf/4193.pdf.

5.3 UTILIZAÇÃO DE MÉTRICAS EM UMA FERRAMENTA DE APOIO A MELHORIA CONTÍNUA DE PROCESSOS DE SOFTWARE

Trabalho de Conclusão de Curso de Pedro Osandy Alves Matos Júnior para obtenção do grau de Bacharel em Ciência da Computação, em 2006, pelo Centro de

Informática da Universidade Federal de Pernambuco em Recife no estado de Pernambuco.

O trabalho contempla a extensão de uma ferramenta de melhoria de processos de software, baseada no modelo IDEAL, para utilização de métricas (MATOS JUNIOR, 2006). E esse trabalho abordou as métricas orientadas a objetivos, ou seja, pelo *Goal Question Metrics* (GQM). O trabalho em questão pode ser encontrado no endereço: http://www.cin.ufpe.br/~tg/2005-2/poamj.pdf.

5.4 UMA FERRAMENTA PARA APOIO À GESTÃO DE ESCOPO DE PROJETOS EM TECNOLOGIA DA INFORMAÇÃO.

Dissertação de Antônio Carlos Marques do Amaral Guerra para obtenção do grau de Mestre em Ciências em 2006 pela Universidade Federal de Uberlândia, através da Faculdade de Engenharia Elétrica em Uberlândia no estado de Minas Gerais.

A dissertação tem como foco o controle dos requisitos que delimitam o escopo do projeto. Para isso foi desenvolvido uma ferramenta que auxilia no dimensionamento do impacto das solicitações de mudanças quanto à duração e o custo do projeto, baseado em dados históricos armazenados no desenvolvimento de projetos anteriores. Para possibilitar a comparação das tarefas com complexidades diferentes, é utilizado o método de métricas de Ponto de Função. Pois entre os métodos de métricas comparados o Ponto de Função foi apontado o melhor método para a comparação das tarefas (GUERRA, 2006). O trabalho em questão pode ser encontrado no endereço: http://www.bdtd.ufu.br/tde_busca/arquivo.php?codArquivo=623

6 ESTUDO COMPARATIVO DE MÉTRICAS DE SOFTWARE PARA UTILIZAÇÃO EM EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE

Este trabalho de pesquisa tem o objetivo de realizar um estudo detalhado dos métodos de métricas de software mais utilizados. Entre os métodos encontram-se as Linhas de Código Fonte, a Análise de Ponto de Função, o Ponto de Caso de Uso e o Modelo de Custo Construtivo. Após o estudo, foi realizada uma pesquisa com a finalidade de analisar qual método de métricas satisfaz melhor ao ambiente de engenharia de software das empresas de desenvolvimento de pequeno porte.

As etapas da metodologia realizadas para a elaboração do trabalho de pesquisa consistiu na busca em livros, revistas e *sites* de trabalhos científicos com conteúdos referentes a fundamentação teórica. Foram realizadas pesquisas nas empresas de desenvolvimento de software de pequeno porte e compatibilidade dos métodos de métricas com a empresa, detalhados na seção seguinte.

6.2 PESQUISA NAS EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE

A fim de estudar detalhadamente os métodos de métricas de software, no qual encontra-se o COCOMO, LOC, APF e UCP, foi realizado uma pesquisa nas empresas de desenvolvimento de software de pequeno porte. O objetivo da pesquisa foi analisar como essas organizações procedem no planejamento e estimativas dos projetos de software.

A forma que foi efetuada a pesquisa consiste em um questionário com os dados para análise de qual método de métricas de software satisfaz melhor o ambiente de engenharia de software da empresa.

Propondo então a empresa utilizar esse método para auxiliar nas estimativas dos projetos e torná-las mais precisas como também melhorar o seu processo de desenvolvimento.

6.2.1 Aplicação do Questionário

O questionário aplicado nas empresas de software foi realizado a partir de requisitos essenciais para obter as seguintes informações necessárias nas entrevistas:

a) a empresa: permite identificar a natureza do negócio da organização;
b) gerenciamento de projetos: identifica a função do gerente de projetos e quais as suas atitudes em um novo software a ser desenvolvido;
c) qualidade do software: verifica como a empresa procura oferecer qualidade do produto aos clientes e a sua equipe e se a empresa possui ou conhece algum certificado;
d) determinação de prazos: permite identificar como essas empresas procedem no cronograma dos projetos e como é distribuída a equipe;
e) monitoração e controle: identifica a forma como a empresa age em caso de alterações nas atividades em execução e se monitoram essas tarefas;
f) análise de riscos: verifica qual a atitude da empresa a frente de um risco e como previnem;
g) informações históricas: permite identificar se a empresa realiza algum armazenamento do que foi realizado nos projetos anteriores;
h) métricas de software: identifica se a empresa utiliza alguma técnica para estimar seus sistemas;

i) estimativas: verifica como estimam os dados do projeto e em que período o faz;

j) programação: permite identificar quais as linguagens de programação utiliza e o estilo que é desenvolvido;

k) modelagem do projeto: verifica se a empresa utiliza a linguagem de modelagem ou protótipos para estruturar o projeto.

Após o levantamento das informações, as questões foram formuladas e organizadas na forma de um questionário semi-estruturado, ou seja, as questões não são formadas objetivamente e é composto pelas questões identificadas no APÊNDICE C e abertas para o entrevistado acrescentar mais informações.

O questionário após a elaboração foi verificado pelo orientador e depois destas etapas feitas e pequenos ajustes, em seguida foi aplicado nas empresas de desenvolvimento de software de pequeno porte.

6.2.2 Processo de Aplicação do Questionário

A pesquisa foi realizada em empresas de desenvolvimento de software de pequeno porte da região carbonífera. Escolhidas por serem mais conhecidas, citadas pelos próprios gerentes de projetos das empresas entrevistas e se disponibilizaram a responder o questionário.

Foram solicitadas entrevistas com dezesseis empresas da região de Criciúma e uma da cidade de Urussanga. Dentre essas empresas, seis tiveram o questionário aplicado através de entrevistas individuais. Ocorreram em salas da própria empresa e de acordo com a disponibilidade de horário do entrevistado. Sendo que cinco empresas o participante respondeu de forma descritiva enviando o resultado por e-mail. Cinco

empresas não aceitaram responder a pesquisa, duas por não poder ceder informações da empresa e três por indisponibilidade de horário por motivo de viagens ou estão empreendendo um novo projeto.

6.3 ANÁLISE DO QUESTIONÁRIO

Ao final do período da pesquisa, várias análises puderam ser feitas a partir dos resultados obtidos do questionário. Na pesquisa considerando sobre o aspecto do gerenciamento de projetos pode-se observar que as empresas entrevistadas apresentam em seus projetos uma coordenação eficiente do gerente, mas que não utilizam ferramentas ou técnicas da engenharia de software para auxiliar no planejamento. Sobre as métricas de software, pode-se notar que uma empresa utiliza as métricas para estimar os projetos, as outras empresas pesquisadas possuem a estrutura para adotar algum método para dar assistência aos gerentes nos planejamentos. A seguir os principais resultados obtidos das entrevistas realizadas com os gerentes de projetos das empresas de desenvolvimento de software.

6.3.1 Gerenciamento de Projetos nas empresas

As empresas de desenvolvimento de software entrevistadas atuam na maior parte delas na área industrial e/ou comercial. Sendo que metade dessas empresas focaliza seus trabalhos em mais de uma área, possibilitando maior abertura de mercado.

Com relação à hierarquia dessas empresas no setor de desenvolvimento de software, ou seja, as divisões de funções consistem em analistas de sistemas e/ou

gerente de projetos e programadores. Os programadores na maioria das empresas se dividem em módulos referentes à especialidade que atua.

Conforme a entrevista os levantamentos das ações que desempenha um gerente de projetos consiste:

a) organizar e fazer cumprir metas;
b) coordenar os trabalhos;
c) realizar o levantamento dos recursos necessários;
d) identificar os riscos possíveis;
e) controlar as atividades;
f) acompanhar as implantações;
g) analisar a aderência do sistema ao cliente;
h) realizar a modelagem de dados e processos;
i) analisar a alocação de recursos pessoal e de ferramentas para manutenção ou desenvolvimento;
j) distribuir as tarefas a sua equipe;
k) analisar qual o objetivo do cliente;
l) monitorar as alterações a serem realizadas;
m) auxiliar a equipe de desenvolvimento;
n) definir interfaces a serem desenvolvidas;
o) definir os custos do projeto.

De acordo com o levantamento, o gerente de projetos é responsável pelo planejamento e também pela continuidade do desenvolvimento do software até completar a sua implantação.

A qualidade nos softwares é uma preocupação para os gerentes de projetos. Pois embora as empresas entrevistadas não possuam certificados de qualidade,

procuram conhecer esses certificados e realizar de alguma maneira a melhoria do ambiente de trabalho e ao produto final.

Para a determinação de prazos são criados etapas ou módulos baseados nos esforços e disponibilidades da equipe. Em algumas empresas essas etapas são formadas pela complexidade e pelo foco do sistema. Após a elaboração é direcionado esse cronograma às equipes responsáveis em que cada membro tem o objetivo de atingir as etapas estabelecidas. Com esse intuito a empresa de desenvolvimento de software pode obter melhor monitoramento das atividades que estão em execução do projeto e corrigir o projeto antecipadamente.

As monitorações do projeto e do desenvolvimento nas empresas entrevistadas são executadas, mas cada uma com o acompanhamento direcionado em um foco. Como por exemplo, algumas monitoram com o objetivo de cumprir as atividades independentes dos prazos, outras empresas controlam os prazos estabelecidos. Caso ocorra alguma alteração no cronograma a maioria das empresas realiza uma análise com todos os *stakeholders*[1] para identificar os motivos e próximas ações a serem efetuadas no projeto.

Os projetos de desenvolvimento de software podem apresentar incertezas que o gerente não teve como prever. Por isso é indispensável a monitoração dos possíveis riscos que poderão apresentar, conforme foi mencionado na análise de riscos na seção 2.3.3. De acordo com a pesquisa efetuada, as empresas realizam o controle de seus projetos e caso ocorra um risco agem de diferentes maneiras, mas há esforços de toda equipe para que resolvam o problema e continuem com as etapas seguintes dos projetos.

[1] *Stakeholder* são todas as pessoas envolvidas e/ou participam do sistema.

Pressman (1995) cita que os gerentes de projetos para desempenharem um projeto de software bem sucedido necessitam se concentrar e ter como prioridade os seguintes elementos-chaves de um projeto: métricas de software, estimativas, qualidade, análise dos riscos, determinação de prazos, monitoração e controle.

Para os quatro últimos elementos-chaves as empresas de desenvolvimento de software entrevistadas apresentam dedicação do gerente de projetos em administrar os projetos com sucesso, mesmo com poucos recursos que utilizam em seus planejamentos. Os dois primeiros elementos-chaves que são métricas de software e estimativas de software serão discutidos a seguir.

6.3.2 Estimativas e métricas de software nas empresas

Estimar os projetos com precisão é um dos maiores desafios no desenvolvimento de software. Pela pesquisa realizada, dez das onze empresas entrevistadas efetuam suas estimativas apenas pela experiência do gerente de projetos. Tornando então a estimativa complexa se o software apresentar características diferentes das anteriores. Porém três dessas empresas além de estimar os projetos por experiência do gerente utilizam as informações e dados dos projetos anteriores para ter um embasamento mais seguro.

Apenas uma empresa utiliza as métricas de software nas estimativas e também como auxílio às informações históricas e a própria a experiência do gerente de projetos. Beneficiando-se assim com melhor estimativas e fornecendo uma visão mais clara da realidade do projeto.

As métricas de software ajudam a determinar a evolução do projeto, melhorar o processo de desenvolvimento e apresentar os dados quantitativos mais

precisos. Porém, para obter na empresa os benefícios das métricas de software no planejamento do projeto, torna-se necessário efetuar uma compatibilidade dos métodos de métricas de software para analisar qual método melhor se adequou ao perfil da empresa.

6.3.3 Compatibilidade dos métodos de métricas de software com as empresas

Para a realização dos estudos foram analisados os métodos de métricas: Linhas de Código Fonte (LOC), Ponto de Caso de Uso (UCP), Ponto de Função (FPA) e COCOMO por serem os mais conhecidos e utilizados.

O método LOC apesar de seu cálculo ser considerado simples, por basear-se apenas em contar as linhas de código fonte do software, apresenta fatores que tendem a não ser aconselhado na utilização em estimativas nas empresas de desenvolvimento de software.

Pode-se citar como desvantagem o tipo de linguagem de programação que a empresa utiliza e o próprio estilo de programação do desenvolvedor, pois influencia na contagem de linhas de código do programa. Entretanto, apesar das empresas pesquisadas padronizarem o estilo de programação, facilitando assim a contagem, as linguagens que utilizam no desenvolvimento são na maioria delas diferentes e dependentes do software a desenvolver. Com isso dificulta a comparação do projeto atual com os softwares anteriores que possuem linguagens diferentes.

Outra desvantagem é que a técnica não tem valor como estimativa no início do planejamento do projeto, uma vez que é necessário realizar a contagem depois da codificação.

O método UCP obtém a vantagem sobre o LOC por ser independente da linguagem utilizada, pois a contagem é realizada em cima dos casos de usos do software. Essa técnica é aconselhada em sistemas desenvolvidos orientados a objetos. Porém esse método obtém algumas desvantagens:

a) impossibilidade de estimativas iniciais do planejamento do projeto devido a ainda não existir a modelagem;
b) a variação na maneira de modelar os requisitos de Caso de Uso pode impactar na contagem obtida;
c) dificuldade de estimar projetos em manutenção nos quais não há modelagem de requisitos;
d) se os critérios utilizados forem muitos diversificados não há garantia que os UCPs estarão medindo a mesma coisa.

Observa-se que o método UCP só poderá ser utilizado caso as empresas adotem a modelagem de caso de uso como forma de mostrar os requisitos. De acordo com a pesquisa, quatro das onze empresas entrevistadas realizam a modelagem, mas destas apenas duas utilizam como estruturação do projeto com uso mais freqüente. Sendo assim o UCP ainda não é uma opção aconselhável para as empresas. Tornando-se então o método Ponto de Função o mais indicado.

O método FPA além de ser facilmente aplicável aos sistemas existentes, pois independe da linguagem de programação, permite realizar a contagem apenas pelo levantamento dos requisitos do sistema. O FPA pode também ser facilmente contado ou estimado a partir de caso de uso se a empresa deseja utilizar na estruturação do projeto. Portanto o Ponto de Função obtém consideráveis vantagens comparadas aos métodos LOC e UCP. Outras vantagens que o Ponto de Função proporciona são:

a) o FPA é padronizado internacionalmente pela ISO, possibilitando uniformidade na aplicação;
b) existência de grande acervo de dados armazenados por diversas organizações possibilitando a realização de estudos e comparações;
c) possibilidade de realizar medições de estimativas no início do planejamento do projeto, na manutenções e alterações ao decorrer do desenvolvimento e no final do projeto;
d) fornece suporte ao gerenciamento de projetos;
e) permite estimativas de produtividade, custos e qualidade com maior acurácia[2];
f) acompanhamento da qualidade e possibilita reestimativas.

Com isso o método FPA considera-se o mais adequado as empresas entrevistadas, pois de acordo com a pesquisa, todas as empresas realizam as estimativas de prazo e custos no início do projeto. Como também o método é independente da tecnologia, torna-se aconselhado já que a maior parte utiliza mais que uma linguagem de programação no desenvolvimento. E conforme entrevista, as empresas têm o hábito de armazenar as informações de seus projetos, facilitando assim a armazenagem das medições realizadas para futuramente calibrar os dados com as novas medições.

Conclui-se então que a aplicação do método FPA nas medidas de estimações é aconselhado, entretanto é preciso que as empresas estejam cientes que a métrica possui limitações. Também é recomendado o cuidado no levantamento dos requisitos e na contagem, pois o cálculo mal realizado pode comprometer o resultado ou até a qualidade do projeto.

[2] Indica a proximidade de uma medição do valor esperado.

O método COCOMO é caracterizado por realizar as estimativas com base no tamanho do software através dos métodos LOC, FPA e UCP. Então não aconselha-se realizar apenas esse método como medições das estimativas, pois já que em seu cálculo utiliza outro método de métrica para a contagem dos dados. Tornando-se então um método complementar às outras três métricas de software.

Independente da métrica utilizada pelas empresas, os métodos não são precisos, mas têm o objetivo de auxiliar os gerentes na tomada de decisão.

CONCLUSÃO

A Engenharia de Software vem se tornando um dos estudos mais abordados quando se contextualiza a evolução no mercado de desenvolvimento de software. Pois o estágio que atualmente se encontra, exige que as empresas de desenvolvimento busquem o aprimoramento de suas técnicas para continuarem competitivas e melhorarem a qualidade de seu produto. Como auxílio aos gerentes de projetos a engenharia oferece as métricas de software para serem seus dados analisados e utilizados nas estimativas.

Nesse trabalho mostrou-se por meio da pesquisa efetuada, como as empresas de desenvolvimento de pequeno porte realizam seu gerenciamento no planejamento de um software. Como também demonstrou um método de métricas de software que auxilia nos processos de um projeto.

A pesquisa realizada apresentou em apenas uma empresa da região carbonífera um método de métricas, as outras mostraram empenho em buscar novas tecnologias para aprimorar suas técnicas. Portanto, se as empresas de desenvolvimento de software quiserem implantar um método para auxiliar no planejamento de seus projetos, possuem todos os requisitos que uma métrica necessita para gerar seus dados.

A pesquisa teve dificuldades em atingir uma dimensão maior do que esperado, pois algumas empresas relutavam em fornecer informações referentes ao gerenciamento do seu projeto. Também houve dificuldades em ampliar a outras regiões além de Criciúma, por questões que levaria mais tempo que o definido no cronograma. Entretanto foi delimitado um número de empresas para realizar uma melhor análise. Contudo a pesquisa foi satisfatória apesar de poucas empresas, pois houve um

entendimento mais específico do funcionamento gerencial de um planejamento de projeto.

O trabalho foi concluído com sucesso, em vista do conhecimento adquirido no final da análise satisfez e mostrou qual o método mais adequado para utilização nessas organizações. O método de métrica de software em questão que foi indicado para as empresas entrevistadas é o método Análise de Ponto Função, por melhor adequar aos requisitos que essas empresas apresentaram na pesquisa.

Como sugestão para trabalhos futuros e continuidade nessa pesquisa sugere-se:

a) aplicar o método de métrica de software proposto nas empresas pesquisadas;
b) realizar a comparação de dois métodos aplicados nas empresas;
c) sugerir métricas mais específicas para cada caso em uma organização;
d) desenvolver uma ferramenta com os métodos implantados;
e) realizar um estudo nos métodos de métricas focados na qualidade do software;
f) analisar o comportamento das métricas de software nas empresas que já utilizam.

REFERÊNCIA

ALDABÓ, Ricardo. **Gerenciamento de Projetos:** procedimento básico e etapas essenciais. 2. ed. São Paulo: Artliber, 2001.

BEZERRA, Cícero Aparecido. **Projetos de Sistemas de Informação baseado em qualidade: uma abordagem voltada à pequena empresa**. 2001. Dissertação (Mestrado em Engenharia de Produção e Sistemas) - Universidade Federal de Santa Catarina. Florianópolis.

BORREGO FILHO, Luiz Fernando. **Uma arquitetura para apoio e automação de processos de gerência de Projetos de Software**. 2003. 304 f. Dissertação (Mestrado do Curso de Pós-Graduação em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais. São José dos Campos.

COUTO, Ana Brasil. **CMMI:** Integração dos Modelos de Capacitação e Maturidade de Sistemas. Rio de Janeiro: Ciência Moderna. 2007.

FIGUEIREDO, Eduardo Magno Lages. **Uma abordagem quantitativa para desenvolvimento de software orientado a aspectos**. 2006. 139 f. Dissertação (Mestrado pelo Programa de Pós-Graduação em Informática) - Pontifica Universidade Católica do Rio de Janeiro. Rio de Janeiro.

GOETS, Carlos Henrique. **Um ambiente para implementação de modelo de gerência de projetos utilizando técnicas de Workflow**. 2003. 90f. Monografia (Tecnólogo em Informática) – Universidade Luterana do Brasil. Canoas.

GRANDCHAMP, Régis Eduardo. **Gerenciamento de Projetos de Software.** Taubaté, 2002. Disponível em: <http://200.136.193.2/prppg/cursos/ppga/mba/2002/grandchamp_regis_eduardo.pdf>. Acesso em: 25 out. 2006

GUERRA, Antonio Carlos Marques do Amaral. **Uma ferramenta para apoio a gestão de escopo de projetos em tecnologia da informação**. 2006. 140 f. Dissertação (Mestrado em Ciências) - Faculdade de Engenharia Elétrica, Universidade Federal de Uberlândia, Uberlândia.

LÓPEZ, Pablo Ariel do Prado. **COCOMO II:** Um modelo para estimativa de custos de Projetos de Software. 2005. 98 f. Monografia (Bacharel em Informática) - Universidade do Vale Do Rio Dos Sinos, São Leopoldo.

MACHADO, Francis Berenger; AMÊNDOLA, Rodrigo Birtel. **Análise Comparativa das Certificações de Qualidade CMM e ISO 9000**: Um Estudo de Caso da IBM Brasil. 2004. Disponível em: <http://www.inf.puc-rio.br/~francis/2004-semead.pdf>. Acesso em: 13 set. 2007.

MARRECO, Daniel Catunda. **Um processo controlável de desenvolvimento de software focado na gestão da qualidade em pequenos projetos**. 2006. 110 f. Dissertação (Mestrado pelo Programa de Pós-Graduação em Informática) – Pontifica Universidade Católica do Rio de Janeiro. Rio de Janeiro.

MATOS JÚNIOR, Pedro Osandy Alves. **Utilização de Métricas em uma ferramenta de apoio à melhoria contínua de processos de software.** 2006. 54 f. Trabalho de Conclusão de Curso (Bacharel em Ciência da Computação) – Centro de Informática da Universidade Federal de Pernambuco. Pernambuco.

MOLINARI, Leonardo. **Gestão de Projetos:** Técnicas e Projetos com ênfase em Web. São Paulo: Érica, 2004.

PERALTA, Antonio Carlos; BULLA, Elisângela Aparecida. **Gestão de Projetos**. 2002. Disponível em: <http://www.dec.uem.br/eventos/enteca2000/artigos/E2000-1-13.pdf>. Acesso em: 20 fev. 2007.

PMI. **A guide to the Project Management Body of Knowledge**. Newtown Square 2000. 3ª ed. Disponível em: <http://www.pmi.org/prod/groups/public/documents/info/pp_pmbokguidethirdexcerpts.pdf>. Acesso em: 23 mar. 2007

PRADO, Darci Santos do. **Gerenciamento de Programas e Projetos nas Organizações**. 3. ed. Nova Lima: INDG tec. S, 2004.

PRESSMAN, Roger S. **Engenharia de Software**. Tradução de José Carlos Barbosa. São Paulo: Pearson Makrom Books, 1995.

SOMMERVILLE, Ian. **Engenharia de Software**. Tradução de André Maurício de Andrade Ribeiro. 6. ed. São Paulo: Addison Wesley, 2003.

STANDISH GROUP. **Chaos Report**. 2004. Disponível em: <http://www.standishgroup.com>. Acesso em: 18 abr. 2007.

VALERIANO, Dalton L. **Gerência em projetos:** Pesquisa, desenvolvimento e engenharia. São Paulo: Makrom Books, 1998.

VAVASSORI, Fabiane Barreto. **Metodologia para o Gerenciamento Distribuído de Projetos e Métrica de Software**. 2002. 211 f. Tese de Doutorado em Engenharia de Produção - Universidade Federal de Santa Catarina. Florianópolis.

WEBER, Kival Chaves; ROCHA, Ana Regina Calvacanti da. **Qualidade e Produtividade de Software**. 3. ed. São Paulo: Pearson Makrom Books, 1999.

YOURDON, Edward. **Declínio e Queda dos Analistas e dos Programadores:** A Salvação e os Novos Caminhos para a produtividade e a Qualidade no Desenvolvimento de Software. Tradução de José Carlos Barbosa dos Santos. São Paulo: Pearson Makrom Books, 1995.

APÊNDICE A – ANÁLISE DE PONTO DE FUNÇÃO

A Análise de Ponto de Função (FPA) é um método utilizado para medições do software em desenvolvimento, visando estabelecer uma medida de seu tamanho sob o ponto de visão do usuário. Para a contagem do ponto de função são realizadas 7 etapas, descritas abaixo.

DETERMINAÇÃO DO TIPO DE CONTAGEM

Os Tipos de Contagem podem ser:

a) projeto: nesse caso a contagem é realizada com o sistema em desenvolvimento;
b) manutenção: a contagem consiste em quando o sistema está sofrendo melhorias e modificações na funções;
c) aplicação: trata-se da contagem em um sistema totalmente desenvolvido.

FRONTEIRAS DE APLICAÇÃO

As Fronteiras de Aplicação consiste na separação do que está sendo contado com aplicações externas. Permitindo identificar o escopo do software e se pertencem ao sistema que está sendo contada.

FUNÇÕES DO TIPO DADO

Chamada também de arquivos de dados (AD) representa as funcionalidades relativas ao requisito de dados internos e externos à aplicação. Classifica a funcionalidade em Arquivos Lógicos Internos (ALI) e Arquivos de interface Externa (AIE). Essas duas categorias se diferem por grupos lógicos relacionados ou informações de controle identificadas pelo usuário mantidas dentro da Fronteira de Aplicação (ALI), ou referenciado pela Fronteira de Aplicação, mas mantido por outro aplicativo (AIE) (GUERRA, 2006).

A contagem da função de dados se baseia na regra de complexidade, em que os elementos devem ser contabilizados como Registros Lógicos Referenciados (RLR) e Dados Elementares Relacionados (DER). Com essa contagem e a contagem de funções do Tipo Transação, obtém-se o valor total de Ponto de Função Não Ajustados (PFNA).

As complexidades funcionais são: Simples, Média e Complexa. Para a identificação do número de funções da complexidade de ALI, utiliza-se como base a Tabela 2.

Tabela 2. Complexidade da ALI

Números de Registros Lógicos	*Quantidade de Itens de Dados*		
	De 1 a 19	***De 20 a 50***	***51 ou mais***
Somente 1	Simples	Simples	Média
2 a 5	Simples	Média	Complexa
6 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A.(2006).

Para traduzir então os ALI em PFNA realiza-se a multiplicação do número de funções simples, médias e complexas pelo peso referenciado na Tabela 3.

Tabela 3. Contagem de Pontos por Função dos ALI

Simples	Média	Complexa
7 PF	10 PF	15 PF

Fonte: BRAGA, A. (1996 apud VAVASSORI, F., 2002).

Para a complexidade funcional dos AIE é realizado da mesma forma. Após a identificação do número de funções, de acordo com a Tabela 4, é realizada a contagem de AIE para PFNA, com a multiplicação do número de funções simples, médias e complexas pelos pesos relacionados na Tabela 5.

Tabela 4. Complexidade da AIE

Números de Registros Lógicos	*Quantidade de Itens de Dados*		
	De 1 a 19	***De 20 a 50***	***51 ou mais***
Somente 1	Simples	Simples	Média
2 a 5	Simples	Média	Complexa
6 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A. (2006).

Tabela 5. Contagem de Pontos por Função dos AIE

Simples	Média	Complexa
5 PF	7 PF	10 PF

Fonte: BRAGA, A. (1996 apud VAVASSORI, F., 2002).

FUNÇÕES DO TIPO TRANSAÇÃO

Representa a funcionalidade provida ao usuário pelo processamento de dados em uma aplicação (VAVASSORI, 2002). Os Tipos de Transação são dividas em três categorias (PRESSMAN, 1995):

a) Entradas Externas (EE): consiste em cada entrada do usuário que proporcione dados distintos orientados à aplicação, ou seja, dados que atravessam a Fronteira de Aplicação de fora para dentro;

b) Saídas Externas (SE): nesse caso os dados são enviados para fora da Fronteira de Aplicação. Podem-se citar como exemplo os relatórios impressos ou mostrados na tela;

c) Consulta Externa (CE): uma consulta é definida como uma entrada do usuário que resulta em alguma resposta de software na forma de saída.

O cálculo da complexidade se baseia no número de arquivos relacionados e Dados Elementares relacionados envolvidos. As complexidades funcionais são: Simples, Média e Complexa. Para a identificação do número de funções da complexidade de EE utiliza-se como base a Tabela 6.

Tabela 6. Complexidade das EE

Números de Arquivos de Dados	*Quantidade de Itens de Dados*		
	De 1 a 4	***De 5 a 15***	***16 ou mais***
0 a 1	Simples	Simples	Média
2	Simples	Média	Complexa
3 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A. (2006).

Em seguida é feito a contagem para PFNA, realizando a multiplicação do número de funções simples, médias e complexas pelos pesos referenciados Tabela 7.

Tabela 7. Contagem de Pontos por Função das EE

Simples	**Média**	**Complexa**
3 PF	4 PF	6 PF

Fonte: BRAGA, A. (1996 apud VAVASSORI, F., 2002).

Para a complexidade da SE é identificado o número de funções de acordo com a Tabela 8, depois é realizado a contagem para PFNA realizando a multiplicação do número de funções simples, médias e complexas pelos pesos referenciados da Tabela 9.

Tabela 8. Complexidade das SE

Números de Arquivos de Dados	*Quantidade de Itens de Dados*		
	De 1 a 4	***De 5 a 15***	***16 ou mais***
0 a 1	Simples	Simples	Média
2 a 3	Simples	Média	Complexa
4 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A. (2006).

Tabela 9. Contagem de Pontos por Função das SE

Simples	**Média**	**Complexa**
4 PF	5 PF	7 PF

Fonte: BRAGA, A. (1996 apud VAVASSORI, F., 2002).

E finalmente para a complexidade da CE determinada observando-se separadamente entradas e saídas. A identificação do número de funções de entrada é feita a partir da Tabela 10.

Tabela 10. Complexidade das CE - Entradas

Números de Arquivos de Dados	*Quantidade de Itens de Dados*		
	De 1 a 4	***De 5 a 15***	***16 ou mais***
0 a 1	Simples	Simples	Média
2 a 3	Simples	Média	Complexa
4 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A. (2006).

Após a identificação de Consultas Externas de Entrada é feito a identificação referente as saídas utilizando como base a Tabela 11.

Tabela 11. Complexidade das CE - Saída

Números de Arquivos de Dados	*Quantidade de Itens de Dados*		
	De 1 a 5	***De 6 a 19***	***20 ou mais***
0 a 1	Simples	Simples	Média
2 a 3	Simples	Média	Complexa
4 ou mais	Média	Complexa	Complexa

Fonte: GUERRA, A. (2006).

Então é realizada a contagem para PFNA, multiplicando o número de funções Simples, Médias e Complexas pelo peso referenciado na Tabela 12.

Tabela 12. Contagem de Pontos por Função das CE

Simples	Média	Complexa
3 PF	4 PF	7 PF

Fonte: BRAGA, A. (1996 apud VAVASSORI, F, 2002).

DETERMINAÇÃO DO PONTO DE FUNÇÃO NÃO AJUSTADO (PFNA)

Realizado o levantamento das funções do Tipo Dados e Tipo de Transação e a sua complexidade funcional é feita a contagem do PFNA. O PFNA fornece uma unidade de produção bruta, a contagem se baseia na seguinte fórmula:

PFNA = ALI + AIE + EE + SE + CE

Onde ALI, AIE, EE, SE e CE é o resultado da soma dos números de funções identificadas de cada complexidade (Simples, Média e Complexa) pelo seu peso de complexidade.

FATOR DE AJUSTE

O Fator de Ajuste (FA) se baseia em 14 Características Gerais do Sistema, citadas na Figura 4, transformando o PFNA em Pontos de Função Ajustados (VAVASSORI, 2002).

1. O sistema requer *backup* e recuperação confiáveis?
2. São exigidas comunicações de dados?
3. Há funções de processamento distribuídas?
4. O desempenho é crítico?
5. O sistema funcionará num ambiente operacional existente, intensivamente utilizado?
6. O sistema requer entrada de dados *on-line*?
7. A entrada de dados *on-line* exige que a transação de entrada seja elaborada em múltiplas telas ou operações?
8. Os arquivos-mestres são atualizados *on-line*?
9. A entrada, saída, arquivos ou consultas são complexos?
10. O processo interno é complexo?
11. O código foi projetado de forma a ser reusável?
12. A conversão e a instalação estão incluídas no projeto?
13. O sistema é projetado para múltiplas instalações em diferentes organizações?
14. A aplicação é projetada de forma a facilitar mudanças e o uso pelo usuário?

Figura 4. Características Gerais dos Sistemas
Fonte: Pressman, R.(1995).

Para determinar os Pontos de Função Ajustado (FA) é realizada a avaliação de cada uma das 14 características gerais pelo Nível de Influência. A graduação do nível de Influência é em uma escala de 0 a 5 pontos, como demonstra a Figura 5.

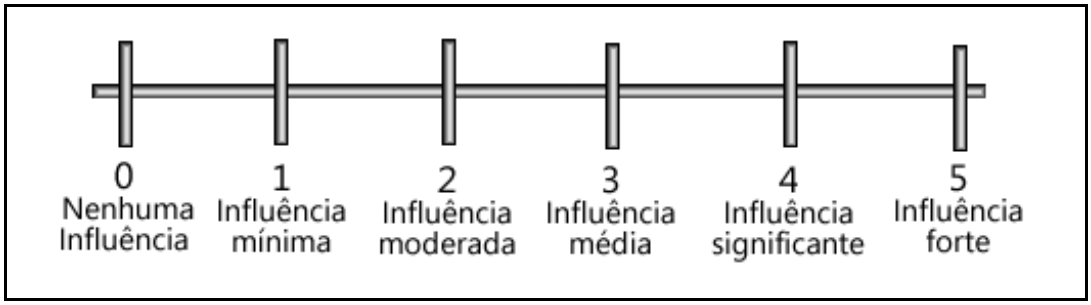

Figura 5. Grau de Influência
Fonte: Pressman, R. (1995).

Somando os níveis de influência de cada característica obtém-se o Nível Total de Influência (NIT). Aplicando o NIT na fórmula, determina-se o valor final do Fator de Ajuste:

$$\mathbf{FA = (NIT * 0{,}01) + 0{,}65}$$

Como o somatório dos graus de influência resultará entre 0 e 70, o FA ajusta o PFNA em mais ou menos 35% (GUERRA, 2006).

PONTO DE FUNÇÃO AJUSTADO

Como última etapa para determinar o Ponto de Função, utiliza-se a fórmula referente ao Tipo de Contagem que foi identificada na primeira etapa:

a) projetos: chamado de Contagem de Ponto de Função de Projetos de Desenvolvimento (PFD) é utilizado em projetos de sistemas em desenvolvimento. A fórmula de Contagem é:

PFD = (PFB + PFC) * FA

Onde:

PFB consiste na Funcionalidade Aplicação onde as funções são obtidas depois da instalação do software;

PFC consiste das funções providas para converterem dados ou necessidade específica de conversão manifestada pelo usuário;

FA é o fator de Ajuste.

b) manutenção: chamado de Contagem de Ponto de Função de Projetos em Manutenção (PFM) é utilizado quando o sistema já está desenvolvido, mas estão sendo realizadas melhorias no software. A fórmula para Contagem é:

PFM = [(INC+ ALT+PFC)*FAD] + (EXC * FAA)

Onde:

INC refere-se aos pontos de funções brutos adicionados a aplicação;

ALT são os pontos de funções alterados na aplicação;

PFC é o ponto de função adicionada pelo processo de conversão;

FAD é o Fator de Ajuste depois da manutenção;

EXC são os pontos de funções brutos excluídos da aplicação;

FAA é o Fator de Ajuste antes da manutenção.

c) aplicação: chamado de Contagem de Ponto de Função de Aplicações Instaladas (PFA) é utilizado em sistemas já desenvolvidos e implantados para o usuário. A fórmula de Contagem é:

PFA = FPNA * FA

Onde:

FPNA refere-se ao ponto de função não ajustado;

FA é o Fator de Ajuste.

APÊNDICE B – PONTO DE CASO DE USO

O Ponto de Caso de Uso (UCP) é realizado para obter a contagem a partir da análise dos casos de uso que compõe o sistema. A métrica UCP é composta de seis etapas descritas a seguir.

VALOR DE IMPACTO DOS AUTORES (UAW - *UNAJUSTED ACTOR WEIGHTS*)

O primeiro passo para obter o UCP é classificar os autores de acordo com a Tabela 13.

Tabela 13. Classificação dos Autores

Tipo de Ator	*Descrição*	*Fator*
Simples	Sistemas Externos	1
Média	Hardware ou temporizadores	2
Complexa	Humanos	3

Fonte: GUERRA, A. (2006).

Após a classificação, para obter o UAW, é realizada a somatória da multiplicação da quantidade final dos tipos de autores pelo fator correspondente.

VALOR DE IMPACTO DOS CASOS DE USO (UUCW - *UNAJUSTED USE CASE WEIGHTS*)

Após o valor do UAW, deve-se classificar o Caso de Uso, de acordo com a Tabela 14.

Tabela 14. Classificação dos Casos de Uso

Tipo de Ator	*Descrição*	*Fator*
Simples	Até 3 caminhos	5
Média	De 4 a 7 caminhos	10
Complexa	Acima de 7 caminhos	15

Fonte: RIBU, K. (2001 apud GUERRA, A., 2006).

Após a classificação, para obter o valor final do UUCW é realizado a somatória da multiplicação da quantidade final do tipo de Caso de Uso pelo fator correspondente.

PONTO DE CASO DE USO NÃO AJUSTADO (UUCP - *UNADJUSTED USE CASE POINTS*)

O cálculo do Ponto de Caso de Uso não Ajustado é uma unidade bruta e precisa ter a influência da complexidade técnica e de ambiente para ser aplicada na estimativa. A fórmula para o cálculo total da UUCP é:

$$\mathbf{UUCP = UAW + UUCW}$$

FATOR DE AJUSTE DA COMPLEXIDADE TÉCNICA (TCF - *TECHNICAL COMPLEXITY FACTOR*)

O cálculo do TCF é obtido com base na Tabela 15 e atribuindo-lhes uma avaliação de 0 a 5 para cada fator.

Tabela 15. Avaliação dos Fatores Técnicos

Fator Técnico	*Peso*
Sistema distribuído	2
Tempo de resposta (desempenho)	1
Eficiência do usuário final	1
Complexidade do processamento interno	1
Código reutilizável	1
Fácil instalação	0,5
Fácil de usar	0,5
Portátil	2
Fácil de mudar	1
Concorrente	1
Características especiais de segurança	1
Acesso direto ao software	1
Necessidade de treinamento especial para o usuário	1

Fonte: RIBU, K. (2001 apud GUERRA, A., 2006).

Após a avaliação é realizada a somatória da multiplicação da avaliação dada pelo fator correspondente para obter o total de Fator Técnico (FT). A fórmula para obter o TCF é:

$$\mathbf{TCF = (0,6 + (0,01* FT)}$$

FATOR DE AJUSTE DA COMPLEXIDADE DE AMBIENTE (EF - *ENVIRONMENT FACTOR*)

São fatores que correspondem a capacidade da equipe. O ECF é obtido com base na Tabela 16, atribuindo-lhes uma avaliação de 0 a 5 para cada fator.

Tabela 16. Avaliação dos Fatores de Ambiente

Fator ambiental	*Peso*
Usando um processo formal de desenvolvimento (metodologia)	1,5
Usuários têm experiência com algum aplicativo anterior	0,5
Experiência orientada a objeto	1
Capacidade do analista chefe	0,5
Motivação	1
Requisitos estáveis	2
Trabalhadores em tempo parcial	-1
Dificuldade de linguagem de programação	-1

Fonte: RIBU, K. (2001 apud GUERRA, A., 2006).

O valor total do Fator Ambiente (EF) é a somatória da multiplicação do valor da avaliação dada pelo peso do fator correspondente. A fórmula para obter o ECF é:

$$\mathbf{ECF = (1,4 + (-0,03* EF)}$$

PONTO DE CASO DE USO AJUSTADO (UCP – *USE CASE POINT*)

Com os componentes UUCP, TCF e ECF já conhecidos, então pode-se calcular o número final do Ponto de Caso de Uso (UCP):

UCP = UUCP * TCF * ECF

APÊNDICE C – QUESTIONÁRIO

Para a realização da pesquisa nas empresas de desenvolvimento de software foi aplicado um questionário com as seguintes questões:

a) Qual o foco da empresa? Os sistemas desenvolvidos são voltados para que área?
b) Como é a hierarquia (divisão de funções) da empresa para a área de desenvolvimento de software?
c) Qual é a função do gerente de projetos? Qual o procedimento do gerenciamento de projetos?
d) A empresa busca a qualidade do ambiente e do produto final? Como? Conhece alguma certificação de qualidade? A empresa possui alguma certificação de qualidade (ISO/IEC, CMM)?
e) Os softwares solicitados são pequenos (exigem poucas pessoas), médios ou complexos(exigem muitas pessoas e recursos)? São em ambientes convencionais (empresas, comércios, etc.) ou complexos (laboratórios de pesquisas, etc.)?
f) Como é feito o cronograma do projeto? Qual o procedimento de divisão das atividades relacionadas no cronograma para a equipe do projeto?
g) Há um acompanhamento das atividades em execução? Qual o procedimento caso ocorra uma alteração no custo, no prazo ou nos cronogramas?
h) Existe uma monitoração para prevenção dos possíveis riscos que podem ocorrer ao longo do desenvolvimento? Como? Qual a atitude da empresa a frente de risco que ocorreu no projeto?

i) A empresa utiliza alguma técnica (métricas de software) para estimar os dados do projeto?

j) Como são realizadas as estimativas de Custo? Prazo? Produtividade do pessoal envolvido?

k) Em que fase do projeto é realizada a estimativa: Logo no início? Quando a estrutura já está definida? Após o desenvolvimento?

l) É realizado o armazenamento das informações dos procedimentos efetuados e as estimativas dos softwares anteriores? Na manutenção desses softwares também?

m) Qual a linguagem de programação que a empresa utiliza?É a mesma para todos os softwares ou depende do sistema?

n) O estilo de programação é padronizado ou o estilo é livre e depende do programador que irá desenvolver?

o) A empresa utiliza alguma linguagem de modelagem (UML) ou protótipo do sistema para estruturar o projeto? Caso utiliza UML, faz logo no início ou depois da estruturação e quais os tipos (diagramas) utilizados?