JAMILLE SILVA MADUREIRA

MAPEAMENTO DAS EMPRESAS DE TI EM VITÓRIA DA CONQUISTA: Uma análise da qualidade do processo de desenvolvimento

Vitória da Conquista – Bahia

Fevereiro - 2007

UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA - UESB
CURSO DE CIÊNCIA DA COMPUTAÇÃO

JAMILLE SILVA MADUREIRA

MAPEAMENTO DAS EMPRESAS DE TI EM VITÓRIA DA CONQUISTA: Uma análise da qualidade do processo de desenvolvimento

Monografia apresentada à disciplina de Projeto de Computação Supervisionado do curso de Ciência da Computação, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientadora: Vanessa Bittencourt X. de Moura

Vitória da Conquista – Bahia

Fevereiro - 2007

JAMILLE SILVA MADUREIRA

MAPEAMENTO DAS EMPRESAS DE TI EM VITÓRIA DA CONQUISTA: Uma análise da qualidade do processo de desenvolvimento

Monografia apresentada à disciplina de Projeto de Computação Supervisionado do curso de Ciência da Computação, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Banca examinadora:

Orientador:

_____
Vanessa Bittencourt Xavier de Moura
Universidade Estadual do Sudoeste da Bahia - UESB

Membro:

_____
Cátia Mesquita Brasil Khouri
Universidade Estadual do Sudoeste da Bahia - UESB

Membro:

_____
Maísa Soares dos Santos
Universidade Estadual do Sudoeste da Bahia – UESB

Vitória da Conquista, 06/02/2007.

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me proporcionado a vida e também por me presentear com a família maravilhosa que tanto amo.

Agradeço aos meus amigos que sempre estiveram do meu lado, me dando força nos momentos difíceis e dividindo as alegrias.

Agradeço aos meus colegas do curso de Ciência da Computação pelos conhecimentos compartilhados.

Agradeço à professora Vanessa Bittencourt, por sempre me atender quando foi preciso e pela excelente orientação.

A todos, muito obrigado!

“Nós cientistas necessitamos especialmente de imaginação. Não bastam a matemática e a lógica: precisamos de um pouco de estética e poesia.”

Maria Mitchell

RESUMO

Os sistemas de computação estão presentes em todos os tipos de empresas. O uso de um software para auxílio nas tarefas é cada vez mais comum. Porém, para que essas tarefas sejam realizadas com sucesso, a qualidade do sistema é fundamental. A garantia dessa qualidade está diretamente ligada ao processo de desenvolvimento. Um bom gerenciamento de projeto envolve a adoção de uma metodologia adequada ao tamanho da equipe e do projeto, planejamento das atividades determinando prazos e responsáveis pelas mesmas, fazer suas respectivas documentações, dentre outras tarefas. Existem padrões de certificação que avaliam a qualidade dos produtos de software. Este trabalho está voltado para aqueles que avaliam a qualidade do processo de desenvolvimento, como por exemplo, CMMI, PSP e MPS-Br. Esses dois últimos estão voltados para empresas de pequeno porte. O objetivo desse trabalho é fazer um mapeamento das empresas de Tecnologia da Informação em Vitória da Conquista. Foi feita uma pesquisa de levantamento, através da aplicação de questionários e pretendeu-se coletar informações sobre os processos utilizados durante o desenvolvimento. Com essas informações, foi feita uma análise da qualidade dos procedimentos das empresas dessa cidade.

PALAVRAS-CHAVE: Processo. Qualidade. MPS-Br. Fábrica de *Software*.

ABSTRACT

The computation systems are present in all of the types of companies. The use of a software for aid in the tasks is more and more common. However, so that those tasks are accomplished with success, the quality of the system is fundamental. The guarantee of that quality is directly linked to the development process. A good project administration involves the adoption of an appropriate methodology to the size of the team and of the project, planning of the activities determining periods and responsible for the same ones, to do their respective documentations, among other tasks. There are certification patterns that evaluate the quality of the software products. This work is gone back to those that evaluate the quality of the development process, as for example, CMMI, PSP and MPS-Br. Those two last they are gone back to companies of small load. The objective of that work is to do a map of the companies of Technology of the Information in Vitória da Conquista. It was made a rising research, through the application of questionnaires and it intended to collect information on the processes used during the development. With those information, it was made an analysis of the quality of the procedures of the companies of that city.

KEYWORDS: Process. Quality. MPS-Br. *Software* Factory.

LISTA DE FIGURAS

Figura 1: Principais fatores da qualidade de produtos de software 21
Figura 2: Evolução da tecnologia da qualidade.....33
Figura 3: Relações entre os atributos internos e externos de *software*.. 34
Figura 4: Os cinco níveis do CMM 37

LISTA DE GRÁFICOS

Gráfico 1: Tempo de atuação das empresas produtoras de *software*.....57
Gráfico 2: Ramo de atuação das empresas produtoras de *software*.....57
Gráfico 3: Segmento de atuação das empresas produtoras de *software*.....57
Gráfico 4: Investimentos.....58
Gráfico 5: Tamanho da equipe de desenvolvimento.....59
Gráfico 6: Escolaridade das equipes das empresas produtoras de *software*.....59
Gráfico 7: Cursos superiores realizados pelas equipes.....59
Gráfico 8: Responsáveis por levantar necessidades do cliente.....60
Gráfico 9: Responsáveis por identificar requisitos.....61
Gráfico 10: Razões da mudança de Requisitos.....61
Gráfico 11: Responsáveis pela elaboração do plano de projeto.....62
Gráfico 12: Responsáveis pela validação do projeto.....63
Gráfico 13: Responsáveis pelo gerenciamento do projeto.....63
Gráfico 14: Responsáveis pela documentação das atividades.....64
Gráfico 15: Responsáveis pela especificação das funcionalidades.....65
Gráfico 16: Responsáveis por elaborar plano de testes.....66
Gráfico 17: Responsáveis por implementar testes.....66
Gráfico 18: Responsáveis por executar testes.....67
Gráfico 19: Responsáveis por executar testes com cliente.....67
Gráfico 20: Responsáveis por visitar o cliente após entrega do sistema.....68
Gráfico 21: Ferramentas de Gerência de Projetos.....69
Gráfico 22: Ferramentas de Modelagem de Sistemas.....69
Gráfico 23: Ferramentas de Controle de Versões.....70
Gráfico 24: Ferramentas de implementação.....70
Gráfico 25: Ferramentas de Banco de Dados.....71
Gráfico 26: Problemas enfrentados pelas empresas.....72
Gráfico 27: Quando prazos são cumpridos.....73
Gráfico 28: Medidas tomadas quando prazos não são cumpridos.....73
Gráfico 29: Cumprimento de custos.....74
Gráfico 30: Quando custos são cumpridos.....74
Gráfico 31: Medidas tomadas quando custos não são cumpridos.....74
Gráfico 32: Realização de tarefas.....75
Gráfico 33: Atividades documentadas.....76
Gráfico 34: Padrões de qualidade conhecidos.....78
Gráfico 35: Padrões de qualidade desejados.....78

LISTA DE ABREVIATURAS

BID – Banco Interamericano de Desenvolvimento

CMM – *Capability Maturity Model*

CMMI – *Capability Maturity Model Integrated*

ETM – Equipe Técnica do Modelo

FCC – Fórum de Credenciamento e Controle

FINEP – Financiadora de Estudos e Projetos

IDEAL– *Initiating Diagnosing Establishing Acting Leveraging*

ISO – *International Organization for Standardization*

MA-MPS – Método de Avaliação do MPS-Br

MCT – Ministério da Ciência e Tecnologia

MPS-Br – Melhoria do Processo de Software Brasileiro

MR-MPS – Modelo de Referência do MPS-Br

PSP – *Personal Software Process*

SOFTEX – Programa Brasileiro de *Software* para Exportação

SPA – *Software Process Assessment*

XP – *Extreme Programming*

SUMÁRIO

1. INTRODUÇÃO 11

1.1 APRESENTAÇÃO DO ASSUNTO 11

1.2 DEFINIÇÃO DO PROBLEMA 11

1.3 OBJETIVOS 12
1.3.1 Objetivo geral 12
1.3.2 Objetivos específicos 12

1.4 JUSTIFICATIVA 12

1.5 METODOLOGIA 13
1.5.1 Tipo de Pesquisa 13
1.5.2 Pesquisa de Apoio 14
1.5.3 Universo e Amostra 14
1.5.4 Instrumentos de Pesquisa 14
1.5.5 Descrição e Análises dos Dados 14

1.6 HIPÓTESES 15

1.7 ESTRUTURA DO TRABALHO 15

2. DESENVOLVIMENTO DE SOFTWARE 16

2.1 CRISE DE *SOFTWARE* 16

2.2 PROCESSOS DE TRABALHO 17
2.2.1 Entendendo o significado de processo 18
2.2.2 Elementos e objetivos do processo 18

2.3 NECESSIDADE DE QUALIDADE DE PROCESSO 20

2.4 PROCESSOS DE DESENVOLVIMENTO 21
2.4.1 Metodologia XP 22
2.4.1.1 Histórico 22
2.4.1.2 Valores 23
2.4.1.3 Práticas 24

2.5 FÁBRICA DE *SOFTWARE* III – Um caso de sucesso no ambiente acadêmico 31

3.1 IMPORTÂNCIA DA QUALIDADE 33

3.2 MÉTRICAS DE QUALIDADE DE *SOFTWARE* 34

3.3 OS DIFERENTES TIPOS DE AVALIAÇÃO 35
3.3.1 CMM 35
3.3.1.2 Os cinco níveis CMM 36
3.3.1.3 Aspectos organizacionais para implantação do CMM 37
3.3.1.4 Melhoria de processos e avaliações segundo o CMM 38
3.3.1.5 Modelos de avaliação e o modelo IDEAL 38
3.3.2 PSP 39
3.3.2.1 Os processos do PSP 39
3.3.2.1 Vantagens do PSP 40
3.3.3 MPS.BR 41

3.3.3.1 Descrição do MR - MPS 42
3.3.3.2 Descrição dos Processos 43

4. MAPEAMENTO DOS PROCESSOS DE DESENVOLVIMENTO EM EMPRESAS PRODUTORAS DE *SOFTWARE* EM VITÓRIA DA CONQUISTA: UMA ANÁLISE DA QUALIDADE 56

4.1 PERFIS DAS EMPRESAS 56
4.2 EQUIPE DE DESENVOLVIMENTO 58
4.3 ATIVIDADES DE DESENVOLVIMENTO 60
4.4 USO DE FERRAMENTAS DE APOIO ÀS ATIVIDADES 68
4.5 CUMPRIMENTO DE PRAZOS E CUSTOS 71
4.6 ANÁLISE DAS ATIVIDADES 75
4.7 METODOLOGIA DE DESENVOLVIMENTO 76
4.8 PADRÕES DE QUALIDADE 77

5. CONCLUSÃO 79

5.1. TRABALHOS FUTUROS 82

6. REFERÊNCIAS 83

APÊNDICES 87

1. INTRODUÇÃO

1.1 APRESENTAÇÃO DO ASSUNTO

É indiscutível a importância do *software* no mundo atual. É comum a presença de um computador para o auxílio de tarefas em quase todos os lugares, como por exemplo, em supermercados, escolas, consultórios médicos, etc.

A garantia que essas tarefas serão executadas com êxito está diretamente ligada à qualidade do *software* em questão. Pressman (1995) define qualidade de *software* como "conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo *software* profissionalmente desenvolvido".

Assim, analisar a qualidade de um *software* não é uma tarefa simples, pois é preciso definir uma métrica ou um conjunto de métricas para um produto não palpável. Sommerville (2003) afirma que a qualidade do *software* está diretamente ligada ao seu processo de desenvolvimento. Desta forma, para analisar a qualidade do *software* deve-se analisar também a qualidade do seu processo.

A proposta desse trabalho é fazer uma análise da qualidade do processo de desenvolvimento seguido pelas empresas produtoras de *software* em Vitória da Conquista-Ba.

1.2 DEFINIÇÃO DO PROBLEMA

A produção de um *software* não é baseada apenas na sua codificação. É necessária a utilização de conceitos de engenharia de *software* e de um processo de desenvolvimento para que o projeto seja bem sucedido. Sommerville (2003) conceitua processo de *software* como "um conjunto de atividades e resultados associados que levam à produção de um produto de *software*".

Para que o resultado final seja satisfatório, é extremamente importante a qualidade do processo de seu desenvolvimento. Mas isso não se resume apenas ao processo em si, sendo relevante os *stakeholders* envolvidos, os produtos de *software*, a estrutura da organização e a satisfação dos clientes e usuários.

Existem alguns padrões específicos para a avaliação da qualidade de *software*, tais como: ISO (*International Organization for Standardization*), CMMI (*Capability Maturity Model Integrated*), PSP (*Personal Software Process*), MPS-BR (Melhoria do Processo de Software Brasileiro) dentre outros. Geralmente, esses padrões são direcionados a organizações mais maduras, não atendendo à realidade de pequenas e micro empresas desenvolvedoras de *software*. Desta forma, essas empresas não seguem um padrão de qualidade e acaba não existindo observação e análise dos produtos gerados.

1.3 OBJETIVOS

1.3.1 Objetivo geral

O objetivo desse trabalho é fazer um mapeamento das empresas de Tecnologia da Informação em Vitória da Conquista fazendo uma análise da qualidade de seus processos de desenvolvimento.

1.3.2 Objetivos específicos

Os principais objetivos dessa monografia são:

1. Identificar as empresas que produzem *software* em Vitória da Conquista.
2. Analisar os processos de desenvolvimento de *software* a fim de obter bases reais do mercado atual nessa área com objetivos científicos;
3. Evidenciar o mercado atual local para instalação de novas empresas produtoras.

1.4 JUSTIFICATIVA

Ao se desenvolver um *software*, a primeira preocupação deve ser a sua qualidade. Um sistema sem esse requisito gera transtornos tanto para a organização quanto para o cliente.

O principal transtorno para o desenvolvedor diz respeito à manutenção do referente *software*. Deverá ser feita uma análise de todo o sistema até descobrir os erros existentes para só depois corrigí-los, o que implica mais tempo e custo financeiro. Além disso, pode-se perder a confiança do cliente.

Já o cliente fica insatisfeito com o serviço, pois, muitas vezes, será obrigado a recorrer à empresa para a reparação de erros, além de correr o risco de suas atividades pararem a depender da importância que o sistema tem na organização.

Por isso, a qualidade do *software* é a sua principal característica, sendo importante uma metodologia que a avalie a partir da definição de métricas. Os padrões mais conhecidos são usados geralmente junto às empresas mais maduras, desestimulando as empresas de pequeno porte a buscarem uma certificação. Assim, os processos utilizados por essas empresas não passam por uma avaliação, como acontece nas empresas locais de Vitória da Conquista.

No momento não há muita competitividade pela falta de visão do mercado em relação à tecnologia da informação. A tendência para os próximos 5 a 10 anos é a valorização da qualidade. A empresa cliente se diferenciará com bons sistemas, pois terão processos organizacionais estáveis.

1.5 METODOLOGIA

A pesquisa realizada para este trabalho foi exploratória e bibliográfica, com objetivo de colher informações suficientes, a fim de se adquirir conhecimentos sobre os processos de desenvolvimento de *software*. Em conjunto, foi realizada uma pesquisa de levantamento, com objetivo de elaborar um delineamento sobre os processos de desenvolvimento de sistema realizados pelas empresas de Vitória da Conquista.

1.5.1 Tipo de Pesquisa

Neste projeto, os procedimentos metodológicos adotados foram a pesquisa bibliográfica e a pesquisa de campo. A metodologia adotada na pesquisa foi exploratória visando proporcionar ao pesquisador conhecer mais sobre o assunto e contribuir para melhor focalização das questões mais importantes, e a metodologia descritiva permite a descrição e correlação das variáveis do projeto.

1.5.2 Pesquisa de Apoio

As pesquisas de apoio foram do tipo bibliográfica, onde foram feitas as coletas dos dados através da pesquisa sobre o assunto em livros, artigos, teses, dissertações entre outros; e de levantamento que foi feito através da aplicação do questionário para a obtenção dos dados requeridos.

.

1.5.3 Universo e Amostra

Para a elaboração desse projeto foi utilizado como universo as empresas de desenvolvimento de *software* na cidade de Vitória de Conquista. A amostra foi composta por oito empresas que se encaixaram nesse perfil e estão situadas nesta cidade. O SEBRAE e JUCEB não têm informações de cadastro dessas empresas. Acredita-se que a amostra de oito empresas é significativa.

1.5.4 Instrumentos de Pesquisa

Para levantar os dados e obter as informações necessárias para o desenvolvimento da pesquisa, foi elaborado questionário com perguntas fechadas com o intuito de atender e alcançar os objetivos da pesquisa, os quais permitiram a construção e conclusão do relatório final. Esse questionário foi aplicado em empresas de desenvolvimento de *software* no município de Vitória da Conquista.

1.5.5 Descrição e Análises dos Dados

As informações e dados coletados através dos questionários aplicados tiveram o auxílio do programa Excel, do pacote da Microsoft Office, que possui facilidade no manuseio, onde os dados foram analisados, tabulados e demonstrados através de gráficos proporcionando uma avaliação mais clara e precisa em termos quantitativos, permitindo assim um resultado mais eficiente do projeto.

1.6 HIPÓTESES

- Não existe preocupação dos metodologistas de padrões de qualidade para avaliação de *softwares* voltados às pequenas e micro empresas;
- As empresas de TI de Vitória da Conquista não possuem um processo definido de desenvolvimento de *software*;
- Não existe controle e acompanhamento da qualidade nas empresas desenvolvedoras de *software* em Vitória da Conquista;
- Sem avaliação da qualidade, hoje, uma empresa desenvolvedora em Vitória da Conquista não é competitiva;
- A tendência é que nos próximos anos o mercado local de produtos de *software* crescerá e será mais competitivo, exigindo maior preocupação com relação aos padrões de qualidade;
- Os egressos do curso de Ciência da Computação são os empreendedores no mercado de Vitória da Conquista de desenvolvimento de *software*.

1.7 ESTRUTURA DO TRABALHO

No primeiro capítulo é apresentada a introdução desta monografia com a descrição do tema com justificativa, problematização e objetivos definidos para este estudo.

O segundo capítulo trata de desenvolvimento de *software*, dificuldades, tarefas das equipes, processo de desenvolvimento e necessidade da qualidade na produção de sistemas de informação.

O terceiro capítulo aborda a importância da qualidade de *software* e os diferentes padrões existentes de avaliação. Trata também da necessidade de um método de fácil aplicação para micro e pequenas empresas desenvolvedoras.

O capítulo quatro mostra a pesquisa de campo realizada em empresas de TI da cidade de Vitória da Conquista. Será feita uma análise dos processos utilizados por essas empresas para desenvolvimento de seus produtos de software.

No quinto capítulo, têm-se a conclusão e a indicação de trabalhos futuros. E, por fim, apresentam-se os anexos, contendo os questionários aplicados e a relação das empresas pesquisadas.

2. DESENVOLVIMENTO DE SOFTWARE

2.1 CRISE DE *SOFTWARE*

O ciclo de desenvolvimento de um *software* passa por diversas dificuldades. Muitas questões devem ser levadas em consideração nesse processo. Cada tarefa que deixa de ser executada ou é feita incorretamente acarretará em problemas mais tarde. O conjunto desses problemas é denominado como crise de *software* (PRESSMAN, 1995).

Esse termo foi utilizado pela primeira vez na década de 70, quando a engenharia de *software* era praticamente inexistente. O termo expressava as dificuldades do desenvolvimento de *software* frente ao rápido crescimento da demanda, da complexidade dos problemas a serem resolvidos e da inexistência de técnicas estabelecidas para o desenvolvimento de sistemas que funcionassem adequadamente ou pudessem ser validados (WIKIPEDIA, 2006).

Envolve problemas reais que são encontrados durante o desenvolvimento de *softwares*. Esses problemas não se referem apenas a um sistema que não funcione corretamente. Pressman (1995) acrescenta: “abrange problemas associados a como desenvolvemos o *software*, como mantemos um volume crescente de *software* existente e como podemos esperar acompanhar a crescente demanda por mais *software*”. O processo de desenvolvimento de sistemas possui uma característica muito peculiar: ele não é visível. Isso leva a alguns problemas que afetarão o sistema final. Um bom exemplo refere-se às estimativas de prazo e de custo que frequentemente são imprecisas.

Assim, alguns projetos podem ter custos excessivos e durar anos. Mesmo quando conseguem ser concluídos podem não ser de qualidade e não satisfazem às necessidades dos clientes.

O *Standish Group Research* mostra que 31.1% dos projetos são cancelados antes mesmo de serem concluídos. Depois é mostrado que 52.7% dos projetos custam 189% a mais das estimativas originais. Em média, apenas 16.2% são completados dentro do prazo e orçamento previstos (THE STANDISH GROUP, 1995)

Como uma alternativa para solucionar esse problema, surge a Engenharia de *Software* que pode ser definida como:

> "disciplina que se ocupa de todos os aspectos da produção de *software*, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação". (SOMMERVILLE, 2003, p. 05)

A engenharia de *software* é composta por um conjunto de atividades que quando executadas levarão a um produto de *software*. Esse conjunto de atividades é denominado de processo de *software*.

2.2 PROCESSOS DE TRABALHO

Apesar das empresas possuírem um processo para produzir bens ou serviços, não existia organização, documentação ou registro desses procedimentos. Ainda hoje, existem algumas empresas que não organizam ou documentam os seus processos. Isto geralmente acontece nas pequenas e médias empresas, pois são construídas pelo seu próprio dono e não tiveram a oportunidade de serem formadas de uma maneira organizada (CRUZ, 1998).

Hoje qualquer erro cometido por uma empresa dá vantagem aos seus concorrentes, pois a competição se tornou cada vez mais feroz. Assim, surgiu a necessidade de analisar e documentar os processos das empresas. Essa atividade é desenvolvida pelo analista de processos, que tem como principais atribuições: entender, levantar, analisar, desenvolver e implantar (GONÇALVES, 2000).

1. Entender:

É importante entender as necessidades do cliente, conhecendo as preocupações de cada usuário envolvido pelo projeto e saber diferenciar as necessidades reais das imaginárias (CRUZ, 1998).

2. Levantar:

Levantar e documentar os dados para entender todas as variáveis relacionadas aos problemas (CRUZ, 1998).

3. Analisar

É feita uma análise do processo de acordo com os dados levantados (CRUZ, 1998).

4. Desenvolver

Desenvolver uma ou várias soluções para o problema. É importante apresentar várias soluções para que o cliente possa escolher (CRUZ, 1998).

5. Implantar

As soluções apresentadas são analisadas pelo cliente, após isso é feita a implantação da escolhida.

2.2.1 Entendendo o significado de processo

Para que tudo isso faça sentido, é preciso explicar o que é um processo. Harrington (1991) define processo como "qualquer atividade ou conjunto de atividades que toma um *input*, adiciona valor a ele e fornece um *output* a um cliente específico".

Assim, um processo tem como função produzir bens ou produtos de uma forma organizada, repetitiva e mantendo a mesma qualidade. Quando é dito "organizada", significa que todo processo deveria ser, mas infelizmente não é o que ocorre nas empresas (CRUZ, 1998).

Um erro comum é a confusão entre processo e procedimento. Procedimento faz parte do processo através de atividades. Outra confusão refere-se a métodos de produção e processos. Método de produção define a técnica pela qual se produz algo enquanto que processo é a forma pela qual essa técnica é empregada (GONÇALVES, 2000).

2.2.2 Elementos e objetivos do processo

Os elementos que compõem os processos são: insumos, recursos, atividades, informações e tempo (CRUZ, 1998).

1. Insumos:

São os fatores que fazem parte de bens ou serviços, como por exemplo, matéria-prima, luz, água, gás, etc. Podem entrar na produção de forma direta ou indireta.

2. Recursos

São eles que dão suporte à produção determinando a capacidade de produzir determinado bem ou serviço. Exemplos de recursos são o capital, mão de obra, equipamentos, etc.

3. Atividades

São as menores partes do processo. Contém dois tipos de informação: identificação que diz a forma da atividade e procedimento que diz a mecânica da atividade.

Os procedimentos, por sua vez, podem ser formais ou informais. Um procedimento é dito formal se foi criado para ser executado pelo ocupante do cargo que corresponde à atividade. Os procedimentos informais são aqueles que não estão escritos em nenhum manual, seja técnico ou funcional.

4. Informações

Definem o processo, dando-lhe a forma e a natureza de sua existência. É um elemento abstrato, o que existe na prática são as atividades.

5. Tempo

É fundamental em qualquer processo, pois o coloca dentro de uma perspectiva prática, informa o que tem que ser feito e quando para a produção de um bem ou serviço.

Os processos contêm objetivos, que são as metas e clientes (GONÇALVES, 2000).

1. Metas

São as definições do que produzir, quantidade, qualidade e em quanto tempo. Devem ser claramente estabelecidas e acordadas com antecedência entre todos os participantes.

2. Clientes

É o principal objetivo de todo processo. Podem ser internos ou externos. Clientes internos são aqueles que desempenham atividades dentro da empresa. Clientes externos são os que vão comprar bens ou serviços produzidos pela empresa.

2.3 NECESSIDADE DE QUALIDADE DE PROCESSO

A qualidade do produto final de *software* está diretamente ligada ao seu processo de desenvolvimento. Assim, o interesse pela melhoria do processo tem aumentado nos últimos anos. Além de melhorar a qualidade do produto, a melhoria do processo pode reduzir custos e tempo de desenvolvimento (SOMMERVILLE, 2003).

Melhorar o processo não é apenas adotar ferramentas ou métodos específicos, ou seguir um modelo que tenha sido utilizado em outro lugar. Existem fatores legais organizacionais, procedimentos e padrões que influenciam o processo e devem ser levados em consideração (SOMMERVILLE, 2003).

Existe uma série de estágios importantes na melhoria de processo, são eles: (SOMMERVILLE, 2003).

1. Análise de processo: envolve examinar os processos existentes e produzir um modelo específico para documentá-los e compreendê-los;
2. Identificação de melhoria: utilizam-se os resultados da análise para identificar gargalos relativos à qualidade, prazo e custo;
3. Introdução de mudança de processo: é a implantação de novos procedimentos, métodos e ferramentas e integrá-los com outras atividades de processo;
4. Treinamento em mudanças de processo: é necessário um treinamento para obter os benefícios das mudanças de processo;
5. Ajuste de mudanças: as mudanças nunca serão totalmente eficazes logo que forem introduzidas. Necessita-se de uma fase de ajuste, onde problemas menores são descobertos e modificações no processo sejam propostas e introduzidas.

Sommerville (2003) afirma que quatro fatores podem afetar a qualidade de um *software*, como mostra a figura 1.

1. Tecnologia de desenvolvimento: quando as equipes são pequenas, a boa tecnologia de desenvolvimento é importante. A equipe não pode gastar muito tempo em procedimentos administrativos;
2. Qualidade do pessoal: quando há poucas pessoas na equipe, a qualidade dos membros é mais importante que o processo de desenvolvimento. Se a equipe for menos capacitada, um bom processo poderá diminuir os danos, mas não

produzirá um sistema com qualidade. Mas isso não significa dizer que ela não deve ter um processo bem definido;

3. Custo, tempo e cronograma: se um projeto tiver um orçamento baixo ou o planejamento do cronograma de entrega não for realista, a qualidade do produto será afetada;
4. Qualidade do processo: para sistemas grandes que são compostos por subsistemas separados, desenvolvidos por diferentes equipes, o que irá determinar a qualidade do *software* será o processo de desenvolvimento. Alguns problemas se referem à integração, gerenciamento de projeto e comunicação entre os membros.

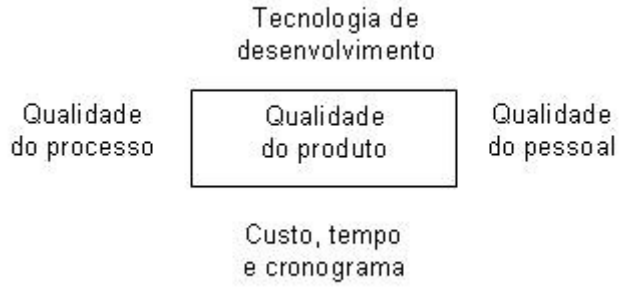

Figura 1: Principais fatores da qualidade de produtos de software
Fonte: Sommerville (2003).

Desta forma, para evitar transtornos durante e após o desenvolvimento de um sistema, é importante que os processos de *software* sejam bem definidos e assim garantir alguma qualidade no produto final.

Existem algumas metodologias que são utilizadas para avaliar a qualidade de *software*, assim como seu processo de desenvolvimento. As mais conhecidas são: ISO, CMMI, MPS-BR e PSP. Os significados dessas siglas foram dados no capítulo 1 desta monografia e as metodologias serão mais detalhadas no capítulo 3.

2.4 PROCESSOS DE DESENVOLVIMENTO

Conforme foi explicado anteriormente, os processos de desenvolvimento de *softwares* são compostos por atividades. Existem tipos diferentes de processos, porém Sommerville (2003) afirma que existem quatro atividades fundamentais presentes em todos eles, a saber:

- Especificação de *software*;
- Projeto e implementação de *software;*
- Validação de *software;*
- Evolução de *software*.

Na especificação são definidos os requisitos do sistema, suas funcionalidades e restrições. Na implementação é quando o *software* é produzido de acordo com o que foi estabelecido na especificação. O sistema é validado em conjunto com o cliente para garantir que ele foi produzido corretamente. Evolução de *software* significa que ele precisa se adequar às novas necessidades do cliente.

Existem alguns modelos de processos genéricos, são abstrações que podem ser utilizadas para explicar diferentes abordagens de desenvolvimento. Mais de um modelo pode ser usado na produção de um sistema. Para a produção de uma parte específica do sistema, um modelo de processo pode ser mais apropriado que outro. O modelo mais conhecido é o desenvolvimento em cascata. Essa metodologia abrange as seguintes etapas: engenharia de sistemas, análise, projeto, codificação, teste e manutenção. Essas atividades são realizadas em seqüência, e a próxima só pode ser executada quando a anterior for concluída (PRESSMAN, 1995).

Com o decorrer do tempo, novas alternativas foram criadas para o modelo em cascata. “Algumas mantiveram parte das características do modelo seqüencial, enquanto outras se baseiam em formas completamente diferentes de se tratar os projetos de *software*” (TELES, 2005).

Um exemplo dessas alternativas é a metodologia *Extreme Programming* mais conhecida como XP. Beck (2000) afirma que a XP “é uma metodologia leve para times de tamanho pequeno a médio, que desenvolvem *softwares* em face a requisitos vagos que se modificam rapidamente”. Assim, esse tipo de modelagem se adeqüa melhor à realidade das empresas locais. Abaixo segue uma apresentação mais detalhada da mesma.

2.4.1 Metodologia XP

2.4.1.1 Histórico

Os princípios fundamentais da *Extreme Programming* se originaram em meados da década de 80, quando Kent Beck e Ward Cunningham trabalhavam na

Tektronixs desenvolvendo sistemas em Smalltalk. Eles faziam uso de práticas como refatoração, programação em pares, mudanças rápidas, entre outras (TELES, 2005).

Alguns trabalhos foram escritos durante dez anos. Esses trabalhos se referiam às boas práticas de desenvolvimento, como por exemplo, o livro *Smalltalk Best Practices Patterns* publicado por Kent (TELES 2005).

Em 1996, Kent foi convidado para analisar o desempenho do sistema C3 (*Chrysler Comprehensive Compensation System)* e foi neste momento que Kent utilizou pela primeira vez, em conjunto, as práticas que atualmente formam a estrutura da XP (TELES, 2005).

A metodologia XP possui quatro valores fundamentais e algumas práticas que guiam o processo de desenvolvimento. Esses valores e práticas serão explicados a seguir (WIKIPEDIA,2006a).

2.4.1.2 Valores

Os principais valores da XP são a comunicação, simplicidade, *feedback* e coragem. (WIKIPEDIA, 2006a).

1. Comunicação:

O processo de desenvolvimento de *software* requer a presença de duas pessoas no mínimo, sendo elas o usuário e o desenvolvedor, causando a necessidade de comunicação. O usuário precisa informar o que ele deseja que seja produzido e o desenvolvedor deve comunicar as considerações técnicas que devem ser levadas em conta (TELES, 2005).

Muitos problemas que ocorreram em projetos foram provocados por alguém que não conversou com outro sobre alguma coisa importante (Beck, 2000).

Uma boa comunicação é essencial para o sucesso do projeto. Uma comunicação ineficaz causará desentendimentos ou compreensão incorreta de algum ponto que poderá afetar o sistema inteiro.

2. Simplicidade:

Um estudo feito pelo *Standish Group* revela que 64 por cento de funcionalidades de um sistema típico poderiam não ter sido implementadas, pelo fato de que 45 por cento delas não são usadas e 19 por cento raramente são.

Assim, os projetos de software freqüentemente investem grande parte dos recursos (tempo, pessoas e dinheiro) em esforços desnecessários (TELES, 2005).

É difícil não antecipar algo que será necessário mais tarde. Porém, “antecipar as coisas compulsivamente é ser influenciado pelo medo do custo exponencial da curva das modificações”, afirmou Benk (2000).

“Os desenvolvedores de um projeto XP procuram implementar as funcionalidades priorizadas para cada iteração com a maior qualidade possível, porém com foco apenas naquilo que é claramente essencial” (JEFFRIES, 2001).

3. *Feedback*:

Uma das tarefas mais importantes em um projeto é a compreensão das necessidades dos usuários. Porém, essa é uma tarefa bem difícil de realizar, pois nem sempre os usuários sabem informa-las corretamente (TELES, 2005).

Assim, uma alternativa para esse problema é o *feedback*, ou seja, um contato constante com o cliente ao longo do projeto, para que as dúvidas sejam tiradas à medida que elas vão surgindo.

4. Coragem:

Existem alguns temores que são inerentes ao processo de desenvolvimento de sistemas. Tanto clientes quanto desenvolvedores têm os seus medos referentes a esse processo. O cliente teme pedir a coisa errada, não obter o que pediu, pagar um preço alto por pouco, etc. Já o desenvolvedor teme não receber definições claras do que precisa ser feito, ser solicitado a fazer mais do que sabe, ser ordenado a fazer algo que não faça sentido, não ter tempo de fazer um bom trabalho,etc (TELES, 2005).

Teles (2005) afirma que “ter coragem em XP significa ter confiança nos mecanismos de segurança utilizados para proteger o projeto.” Sabe-se que problemas irão ocorrer, entretanto, a equipe utiliza métodos que possam ajudar a reduzir ou eliminar as conseqüências destes problemas.

2.4.1.3 Práticas

A XP possui treze práticas que são utilizadas no processo de desenvolvimento de sistemas, sendo elas: cliente presente, jogo do planejamento,

programação em pares, reuniões em pé, código coletivo, código padronizado, desenvolvimento orientado a testes, design simples, refatoração, *releases* curtos, integração contínua, metáfora e ritmo sustentável (WIKIPEDIA, 2006a).

1. Cliente presente:

Um sistema para ser produzido precisa da participação tanto de desenvolvedores quanto de clientes. A equipe de desenvolvimento precisa de uma especificação mais clara do sistema e o cliente é quem deve passar essas informações (TELES, 2005).

O contato com o cliente de forma contínua é importante porque:

> "a presença do cliente ao longo do desenvolvimento viabiliza o ciclo contínuo de *feedback* entre ele e os desenvolvedores. Este ciclo permite que pequenas mudanças sejam feitas ao longo do desenvolvimento, de forma rápida." (TELES, 2005, p. 72)

O problema é que nem sempre o contato pessoal acontece de uma forma fácil, e às vezes ele nem é possível. Assim, torna-se necessária a criação de alternativas que diminuam a distância entre desenvolvedores e clientes (TELES, 2005).

2. Jogo do planejamento

Teles (2005) afirma que uma das tarefas mais importantes em um projeto é decidir o que vai ser implementado. O desenvolvimento é feito em iterações semanais. No início de cada semana, desenvolvedores e clientes reúnem-se para definir e priorizar as funcionalidades. Durante essas reuniões o cliente identifica prioridades e os desenvolvedores as estimam. Como o escopo é reavaliado semanalmente, o projeto é regido por um contrato de escopo negociável, que difere significativamente das formas tradicionais de contratação de projetos de software. Ao final de cada semana, o cliente recebe novas funcionalidades, completamente testadas e prontas para serem postas em produção (WIKIPEDIA, 2006a).

Teles apud Beck e Fowler (2001, p.21) explica os conceitos de iteração e release: "O XP tem releases que tomam alguns poucos meses, que se dividem em iterações de duas semanas, que se dividem em tarefas que tomam alguns dias".

O planejamento é usado em XP para assegurar que a equipe esteja sempre trabalhando no mais importante, a cada momento do projeto (TELES, 2005).

3. Programação em par

Williams e Kessler (2003) definem programação em par como “um estilo de programação no qual dois programadores trabalham lado a lado em um computador, continuamente colaborando no mesmo design, algoritmo, código e teste.”

Normalmente a dupla é composta por um iniciante na linguagem e a outra pessoa funciona como um instrutor. Pelo fato de usarem apenas um computador, o novato fica à frente fazendo a codificação, enquanto o instrutor acompanha ajudando a desenvolver suas habilidades. Assim o programa é sempre revisto por duas pessoas, evitando e diminuindo assim a possibilidade de erros. Com isto busca-se sempre a evolução da equipe, melhorando a qualidade do código fonte gerado (WIKIPEDIA, 2006a).

Essa prática torna o processo de inspeção parte natural do dia-a-dia da equipe de desenvolvimento. Assim ela utiliza inspeções com freqüência sem sofrer os problemas que tornam as inspeções tradicionalmente desagradáveis (TELES, 2005).

4. Reuniões em pé

Com o objetivo de assegurar que as partes trabalhem bem em conjunto, todos os dias a equipe se reúne rapidamente, com todos os membros em pé. Eles informam os resultados obtidos no dia anterior permitindo que os participantes priorizem as atividades do dia que se inicia (TELES, 2005).

Essas reuniões forçam uma aproximação maior entre os desenvolvedores. Assim, toda a equipe possui um conhecimento dos desafios que foram enfrentados por cada membro, das soluções criadas, das idéias colocadas em prática e dos problemas que precisam ser tratados com urgência (TELES, 2005).

5. Código coletivo

Outra prática da XP é que o código é de propriedade coletiva, ou seja, o código pertence à equipe desenvolvedora, e não apenas quem o criou, assim, qualquer membro da equipe pode melhorar o que for necessário (TELES, 2005).

Assim, o código fonte não tem dono e ninguém precisa solicitar permissão para poder modificar o mesmo. O objetivo é fazer a equipe conhecer todas as partes do sistema (WIKIPEDIA, 2006a).

É importante ressaltar que alterações no código podem causar erros. Portanto, estas práticas só podem ser adotadas com segurança se a equipe investir em testes automatizados (TELES, 2005).

Uma vantagem dessa prática é que a equipe se torna mais robusta, pois os desenvolvedores trabalham nas mais variadas partes do sistema.

Sendo assim:

> "todo o trabalho fica menos suscetível de ser afetado se alguém ficar doente (...) ou faltar (...). Isso não apenas reduz as variações no cronograma, mas também eleva as chances de ter alguém por perto quando o código tiver que ser modificado "
> Teles apud Weinberg (1971, p.59-60)

Essa prática também permite que a equipe avance mais rapidamente, pois ninguém precisa esperar que outra pessoa apareça para reparar algum erro (TELES, 2005).

6. Código padronizado

Antes da codificação do sistema, os programadores estabelecem um padrão que deverá ser adotado durante a implementação. O importante é que todos os membros da equipe adotem esse padrão. (JEFFRIES, ANDERSON et al., 2001).

Isso ajuda a simplificar a comunicação entre os membros da equipe e também a utilização de outras duas práticas: a programação em par e o código coletivo. Como o código passa por vários níveis de revisão, é fácil identificar e corrigir um código fora do padrão (TELES, 2005).

Para assegurar que o padrão seja utilizado, "qualquer padrão envolvido deve surgir e evoluir no nível das próprias pessoas que realizam o trabalho. A propriedade do padrão deve estar nas mãos daqueles que realizam o trabalho" (DEMARCO, 2001).

7. *Design* Simples

Para garantir que o sistema esteja funcionando corretamente, o desenvolvimento é executado em iterações curtas onde se implementa um pequeno

conjunto de histórias. O *feedback* gerado ao final de cada iteração permite que os usuários avaliem o sistema. (TELES, 2005)

Adotar iterações curtas é um mecanismo importante para garantir que o *software* faz o que o cliente deseja, entretanto, impõe um desafio. Os desenvolvedores têm uma ou duas semanas para fazer a análise, design, implementação, teste e depuração de um conjunto de funcionalidades. Isso se torna possível se os desenvolvedores mantiverem o design da aplicação suficientemente simples (BECK, 2000).

8. Desenvolvimento orientado a testes

O processo de desenvolvimento de *softwares* passa por diversas dificuldades ao longo do tempo. Um dos problemas mais caros e recorrentes é a ocorrência de erros. Quando um defeito é identificado, faz-se necessário depurar o software. “Depuração é a parte difícil e lenta da programação de sistemas, e ciclos de *feedback* demorados representam o pior problema da depuração (BROOKS, 1995)”.

É essencial que as equipes de desenvolvimento sejam capazes de reduzir a incidência de erros e o custo associado à depuração e eliminação dos mesmos. Isso é válido durante o projeto, porém é ainda mais relevante durante a manutenção do sistema.

A prática de desenvolvimento orientado a testes consiste em escrever um mecanismo de teste automatizado antes de codificar cada história e cada método do sistema (TELES,2005). É uma técnica preventiva utilizada durante o projeto e a manutenção do sistema. A prevenção é usada porque “(...) antes da questão de como resolver problemas vem a questão e *evitar* problemas. (...) um programador que evita problemas é mais ‘inteligente’ que aquele que traz problemas para si mesmo” (WEINBERG, 1971).

Os testes automatizados servem para comprovar que as solicitações do cliente estão sendo atendidas. “De tudo o que podemos esperar de um programa, primeiro e mais importante, é que ele esteja correto.” (WEINBERG, 1971).

Existem dois tipos de testes: o de unidade e o de aceitação. O teste de unidade tenta assegurar que cada componente do sistema funcione da forma correta enquanto que o de aceitação procura testar a interação entre os componentes, as quais dão origem às funcionalidades (BECK, 2000).

9. Refatoração

Equipes XP investem diariamente no aprimoramento do design e na identificação de pontos da arquitetura que estejam se degradando.

A “refatoração é o processo de fazer mudanças em um código existente e funcional sem alterar seu comportamento externo. Em outras palavras, alterar *como* ele faz, mas não *o que* ele faz. (ASTELS, 2003, p.15).”

Cada passo da refatoração de um código é simples, apesar disso, o efeito cumulativo destas pequenas mudanças podem melhorar significativamente o design.

Os desenvolvedores procuram evitar situações nas quais os programas se tornem difíceis de serem modificados. Um sistema mal desenvolvido é difícil de alterar. Assim, o objetivo é elaborar programas que “sejam fáceis de ler, tenham toda a lógica em apenas um lugar, não permitam que mudanças ameacem o comportamento existente e permitam que lógicas condicionais sejam expressas da maneira mais simples possível (FOWLER, 2000, p.60)”.

10. *Releases* curtos

Releases curtos significa colocar o sistema em produção com freqüência e em prazos curtos (dois ou três meses na média). Isso é bom porque “clientes gostam de entregas rápidas. (...) entrega rápida normalmente se traduz em aumento de flexibilidade no negócio (POPPENDIECK, POPPENDIECK, 2003, p.70-71)”.

Entrega rápida possibilita que mantenha as opções em aberto até que tenha reduzido incertezas e possa tomar decisões mais informadas e baseadas em fatos (POPPENDIECK, POPPENDIECK, 2003, p.70-71).

A equipe procura garantir que os primeiros *releases* gerem a maior parte do valor do sistema porque geralmente eles contêm as funcionalidades com maior valor para o negócio. Com *releases* em produção rápida, o cliente tem a oportunidade de receber *feedback* concreto sobre o real potencial de retorno do projeto. Assim, o cliente tem a chance de decidir cedo se continua ou não com o projeto e na mesma equipe de desenvolvimento. Desta forma, a adoção de releases curtos funciona como uma estratégia de gestão de risco do projeto (TELES, 2004).

11. Integração contínua

As equipes normalmente são formadas por diversos programadores que trabalham em pares conforme a prática de código coletivo. Assim, surgem dois

problemas. O primeiro é que “sempre que diversos indivíduos estão trabalhando na mesma coisa, ocorre uma necessidade de sincronização (POPPENDIECK, POPPENDIECK, 2003)”. O segundo é que os pares precisam ser capazes de evoluir rapidamente sem interferir no trabalho uns dos outros. (BECK, 2000).

Esta questão é resolvida utilizando-se uma prática conhecida como integração contínua. Sempre que produzir uma nova funcionalidade, não se deve esperar uma semana para integrar à versão atual do sistema. Isto aumenta a possibilidade de conflitos e a possibilidade de erros no código fonte. Integrar de forma contínua permite saber o status real da programação (WIKIPEDIA, 2006a).

12. Metáfora

Uma equipe de desenvolvimento tem o desafio de manter a integridade conceitual do sistema mesmo havendo diversos projetistas criando estruturas novas na arquitetura ao longo do tempo (TELES, 2005).

A solução é alinhar o pensamento dos projetistas assegurando que todos compartilhem uma visão única de como adicionar e manter as funcionalidades do sistema. Para isso, a XP faz uso do conceito de metáfora (TELES, 2005).

Outro problema que a metáfora resolve é a comunicação com o cliente. Muitas vezes ele não sabe expressar corretamente os requisitos do sistema, portanto, é preciso traduzir as palavras do cliente para o significado que ele espera dentro do projeto. (WIKIPEDIA, 2006a).

13. Ritmo sustentável

Recomenda-se que os desenvolvedores trabalhem apenas durante o tempo regulamentar, ou seja, oito horas por dia, e evitem horas-extras sempre que possível. Sugere-se que os prazos sejam mantidos fixos e as cargas de trabalho sejam ajustadas (através de priorização) para se adequar aos prazos (FOWLER, 2001).

Além da XP, as empresas de pequeno porte podem optar pela adoção da Fábrica de *Software* III, que é um processo onde a organização passa a trabalhar de uma maneira mais sistematizada. A próxima seção apresentará a Fábrica III, tratando de seu histórico e fases do processo de desenvolvimento de sistemas.

2.5 FÁBRICA DE *SOFTWARE* III – Um caso de sucesso no ambiente acadêmico

O processo da Fábrica de *Software* III foi definido e aplicado no ambiente acadêmico da Universidade Federal de Pernambuco (UFPE) no ano de 2003. Porém, o seu conceito não é novo, ele surgiu no final da década de 60, nos Estados Unidos. O termo deve ser interpretado como uma organização projetada de forma sistematizada onde as tarefas são organizadas e a padronização é utilizada com o objetivo de auxiliar na coordenação e formalização do processo (ROCHA apud AAEN; BOTTCHER; MATHIASSEN, 1997).

A metodologia de desenvolvimento de *software* da Fábrica III é composta por quatro fases: comercial, planejamento e gerenciamento, desenvolvimento de componentes e testes e validação. Em cada uma dessas fases existem os responsáveis por gerar artefatos que servirão como base para analisar o progresso do projeto (FÁBRICA DE *SOFTWARE* III).

A fase comercial é a da definição do projeto, de acordo com as necessidades levantadas junto ao cliente. Essas necessidades são detalhadas em requisitos funcionais e não-funcionais, além de serviços associados. O custo e esforço são estimados e apresentados ao cliente, que decide se contrata ou não o projeto. Independente da decisão, a fase é terminada nesse momento (FÁBRICA DE *SOFTWARE* III).

A fase do planejamento consiste em entender o produto a ser desenvolvido e fornecer uma estrutura que possibilite fazer estimativas razoáveis de recursos, custos e prazos. O gerenciamento da evolução do projeto é outro passo importante, uma boa forma de acompanhamento é através de controle; logo, deverão ser desenvolvidos mecanismos para avaliar o progresso, organizar o pessoal que desenvolverá o produto, rastrear e controlar o projeto (FÁBRICA DE *SOFTWARE* III).

Durante a fase de desenvolvimento, as tarefas a serem realizadas pelo analista de sistema são estudar o problema em questão, definir requisitos e projetar o sistema. O engenheiro de software é responsável por realizar a codificação dos componentes projetados pelo analista de sistemas. (FÁBRICA DE *SOFTWARE* III).

A última etapa é a de testes e validação. Durante o processo de desenvolvimento de software há uma grande possibilidade de ocorrência de falhas no produto. O custo da correção destas falhas é muito alto. Por este motivo, uma

série de testes bem executados, iniciados no começo do ciclo de vida do software, reduz significativamente o custo com manutenção de software. Os testes representam a última oportunidade de detectar erros antes do software ser distribuído aos usuários. Depois da realização de testes é preciso validar o sistema com o cliente para que ele possa entrar em operação (FÁBRICA DE *SOFTWARE* III).

O acompanhamento detalhado das tarefas e suas respectivas documentações são importantes para garantir qualidade no processo de desenvolvimento.

3. MÉTODOS PARA AVALIAÇÃO DA QUALIDADE

3.1 IMPORTÂNCIA DA QUALIDADE

Chiossi e Côrtes (2001) explicam que, após a Segunda Guerra Mundial, aumentaram a capacidade de produção e a diversidade dos produtos. Como conseqüência, aumentou-se também a competitividade entre as organizações. Assim, o diferencial entre elas é justamente a qualidade.

Desta forma, a importância da qualidade é essencial para uma empresa que deseja se manter no mercado. Sommerville (2003) diz que a maioria das organizações tem como objetivo principal atingir um alto nível de qualidade no seu produto ou em seus serviços.

No decorrer dos anos, surgiram três tecnologias de avaliação da qualidade. A primeira foi a inspeção no produto. Nesta tecnologia, os produtos intermediários e finais eram examinados para se detectarem os defeitos. Em seguida surgiu o controle de qualidade, onde as taxas de defeitos e os custos são monitorados com o objetivo de identificar os elementos defeituosos dos processos e posteriormente corrigi-los. A tecnologia atual é a melhoria de processos. Ela busca melhorar o processo de produção para minimizar a introdução de defeitos, diminuindo assim o custo. A figura 2 mostra como essas tecnologias evoluíram através do tempo (CHIOSSI, CÔRTES, 2001).

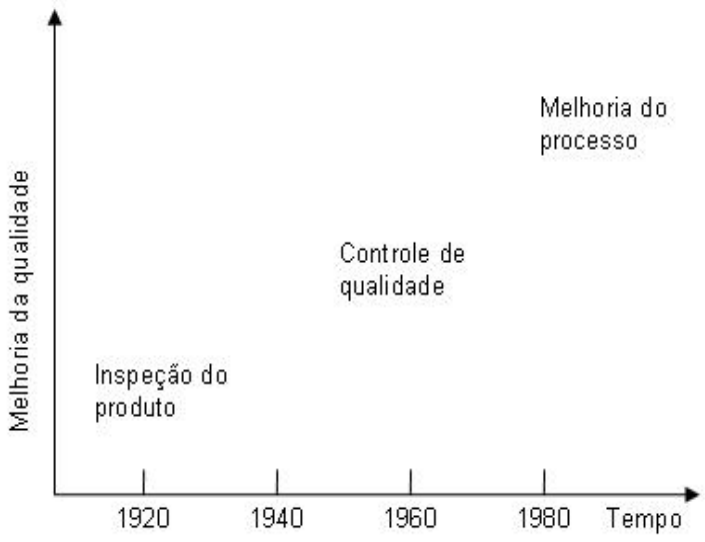

Figura 2: Evolução da tecnologia da qualidade
Fonte: CHIOSSI, CÔRTES, 2001.

3.2 MÉTRICAS DE QUALIDADE DE *SOFTWARE*

No contexto de desenvolvimento de software, qualidade pode ser entendida como um conjunto de características a serem satisfeitas em um determinado grau, de modo que o produto de software atenda às necessidades explícitas e implícitas de seus usuários (DUARTE, FALBO, 2000).

Medir um *software* significa “obter um valor numérico para alguns atributos de um produto ou de um processo de *software*” (SOMMERVILLE, 2003).

O uso de métricas beneficia tanto desenvolvedores quanto clientes. A determinação do nível de qualidade desejada durante os testes, permite que os desenvolvedores avaliem o impacto de diferentes ações sobre a qualidade. Os usuários também se beneficiam com o uso de métricas, pois elas estão diretamente relacionadas à eficiência de operação do sistema (CHIOSSI, CÔRTES, 2001).

Os atributos externos do *software* (confiabilidade, portabilidade, facilidade de manutenção e uso) são afetados por muitos fatores diferentes e não existem métricas simples e diretas para eles. Assim, é preciso medir os atributos internos (número de linhas de código e de erros, complexidade ciclomática, etc.) e supor a relação com os externos para que estes sejam avaliados (SOMMERVILLE, 2003). A figura 3 mostra algumas relações entre os atributos internos e externos.

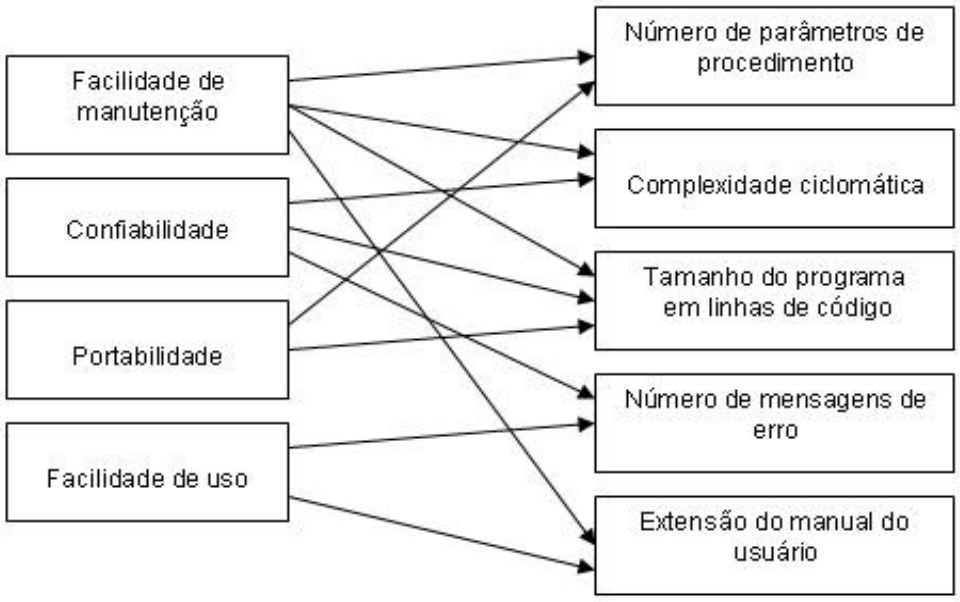

Figura 3: Relações entre os atributos internos e externos de *software*..
Fonte: Sommerville (2003).

Sem a aplicação de métricas, o estado do *software* está sujeito à subjetividade, por isso elas são tão importantes. Através delas que é possível analisar a qualidade dos sistemas (DUARTE, FALBO, 2000).

3.3 OS DIFERENTES TIPOS DE AVALIAÇÃO

3.3.1 CMM

O CMM (*Capability Maturity Model*) foi desenvolvido no SEI em 1993. Verificou-se nas pesquisas nesse instituto que a grande maioria das organizações não possuíam o mínimo de organização, gerenciamento e eficiência nas atividades de desenvolvimento de software. Dessa preocupação surgiu o propósito principal do CMM que é ajudar organizações a conhecerem e melhorarem seus processos de desenvolvimento e manutenção de *software* (SANTANDER, VASCONCELOS, 2000)

Esse modelo tem como foco a maturidade dos processos de *software* e é o mais usado. Os conceitos básicos do CMM podem ser usados tanto para melhoria quanto para avaliação de processos. O CMM possui as seguintes propostas: (JOSKO, CÔRTES, 2005).

a) ser baseado em experiência prática de empresas de *software*;
b) refletir o melhor do estado da prática;
c) atender as necessidades daqueles que realizam melhoria do processo e avaliação do processo de *software*;
d) ser documentado e estar disponível publicamente.

Para perceber as vantagens que uma empresa pode ter engajando-se num programa de evolução de maturidade, é preciso entender as diferenças entre as empresas maturas e imaturas.

A principal característica em uma empresa imatura é que os processos de desenvolvimento são improvisados, ou se existem, não são seguidos. Outras características são: o trabalho é feito em regime de emergência, dificilmente os compromissos de prazo e custo são cumpridos, não é hábito planejar com base em estimativas realistas e quando o projeto é pressionado por prazo, as características de qualidade e funcionalidades do produto são sacrificadas (CHIOSSI, CÔRTES, 2001).

3.3.1.2 Os cinco níveis CMM

As práticas do modelo CMM são organizadas em cinco níveis: inicial, repetível, definido, gerenciado e em otimização.

No nível inicial, não existe repetição dos processos e frequentemente assume compromissos que não consegue cumprir, seja em prazo ou em custo. Não existe planejamento das atividades, as tarefas são executadas na medida em que vão surgindo em caráter emergencial. As chances de sucesso dependem de habilidades pessoais de alguns membros da equipe, e principalmente de atitudes “heróicas”. Nesse nível, as qualidades, os procedimentos e conhecimento pertencem às pessoas e não ao projeto, assim, a capacidade do processo não pertencem à equipe (JOSKO, CÔRTES, 2005).

No nível repetível, as políticas e procedimentos essenciais durante o gerenciamento do desenvolvimento de *software* estão definidos e são obedecidos. A experiência anterior em projetos similares é usada como base para os novos projetos. Os compromissos são assumidos de acordo com bases na experiência adquirida e nos requisitos devidamente documentados. Existe um planejamento das atividades, assim, o desenvolvimento é acompanhado e os planos são revisados regularmente. Esses planos incluem prazos, custos e as funcionalidades do *software*. Os processos afetados pertencem à equipe e são de natureza gerencial e não técnicos (SANTANDER, VASCONCELOS, 2000).

No terceiro nível, o definido, os processos são estabelecidos e padronizados para toda a organização. Os processos de engenharia de *software* são considerados junto dos processos gerenciais. Todo o pessoal técnico e gerencial passa por um treinamento para garantir que ele tenha os conhecimentos e habilidades necessárias para desempenhar suas funções. Nessa fase, os processos não pertencem mais às equipes, mas sim à organização (CHIOSSI, CÔRTES, 2001).

No nível gerenciado, metas quantitativas para os produtos e processos são estabelecidas pela organização. Para que possa ser feita a análise do desempenho, medidas de qualidade e produtividade são coletadas em todos os projetos e armazenadas numa base de dados. A organização passa a ter uma gestão em base quantitativa (JOSKO, CÔRTES, 2005).

No quinto e último nível, em otimização, a organização está envolvida na melhoria contínua de seus processos. É capaz de identificar os erros e agir antes que eles aconteçam. É possível executar ações com o objetivo de evitar retrabalho e desperdício de custos e tempo, melhorando a produtividade. Melhorias de processos e tecnologia são planejadas e executadas como atividades de rotina da organização (CHIOSSI, CÔRTES, 2001).

A figura 4 ilustra os cinco níveis do CMM:

Figura 4: Os cinco níveis do CMM
Fonte: Chiossi, Côrtes (2001, p. 68).

3.3.1.3 Aspectos organizacionais para implantação do CMM

A proposta do CMM é ser independente do tipo de estrutura organizacional, porém, em algumas práticas são mencionadas responsabilidades de pessoas e grupos. Esse modelo cita três perfis funcionais, sendo que em empresas de pequeno porte, uma pessoa pode assumir mais de um perfil (JOSKO, CÔRTES, 2005).

- gerentes: podem possuir as seguintes funcionalidades: responsabilidade pelo projeto, responsabilidade pela gerência de alto nível e também a responsabilidade pela infra-estrutura;
- líder: responsável por coordenar a equipe com tarefas no ciclo de desenvolvimento;

- desenvolvedores: responsáveis pela codificação do sistema.

Existem ainda os grupos que são formados por pessoas que irão desenvolver atividades semelhantes. Alguns exemplos são:

- grupo de engenharia de *software*: as pessoas são os responsáveis pela condução das atividades de desenvolvimento e manutenção do sistema;
- grupo de teste: o responsável pela fase de teste do *software*;
- grupo de configuração de *software*: as pessoas desse grupo são responsáveis pelas atividades de configuração do sistema.

3.3.1.4 Melhoria de processos e avaliações segundo o CMM

Antes de começar a mudar características de uma empresa, algumas atividades são necessárias, para que se obtenha o resultado desejado com a mudança. A primeira atividade é conhecer a empresa, saber em que situação ela se encontra no momento atual. Assim, define-se a situação em que a empresa quer se encontrar após a mudança. Assim o planejamento das ações que serão tomadas poderá ser feito com mais clareza. O modelo de melhoria IDEAL auxilia na execução dessas tarefas. (SANTANDER, VASCONCELOS, 2000).

3.3.1.5 Modelos de avaliação e o modelo IDEAL

O CMM possui dois tipos de avaliação: o primeiro é denominado de SPA (*Software Process Assessment*) que avalia o processo de *software* e é usado na determinação da situação atual dos processos de *software* com a finalidade de melhorá-los; o outro é o *Software capability evaluation*, que avalia a capacidade dos processos de *software* e é usado para avaliar subcontratados ou futuros fornecedores de *software* (JOSKO, CÔRTES, 2005).

O modelo IDEAL (*Initiating Diagnosing Establishing Acting Leveraging*) apresenta “o ciclo de melhoria desde a identificação da necessidade até a institucionalização e estabelecimento do programa de melhoria contínua”. (CHIOSSI, CÔRTES, 2001, p. 80).

3.3.2 PSP

Os outros modelos de qualidade se preocupam ou com o produto (como por exemplo, a ISO/IEC 9126) ou com o processo de desenvolvimento (CMM). O problema dessas abordagens é que elas têm difícil aplicação em empresas de pequeno porte, onde a equipe de desenvolvimento é pequena. O PSP (*Personal Software Process*) surgiu para tentar resolver essa questão, onde a preocupação está voltada ao indivíduo (CHIOSSI, CÔRTES, 2001).

O principal objetivo do PSP é estabelecer um mecanismo para melhorar a capacidade de planejamento, acompanhamento e qualidade dos resultados, mas com foco no nível pessoal. Seus conceitos básicos podem ser usados como ferramenta de uso para gerenciamento das atividades pessoais (WILSON JÚNIOR,2000)

A proposta é que haja uma integração entre o PSP e as práticas organizacionais do CMM, permitindo que os desenvolvedores tenham um melhor desempenho pessoal. Para que isso seja possível, é necessário conhecer, controlar e melhorar os processos usados no nível pessoal. (CHIOSSI, CÔRTES, 2001).

3.3.2.1 Os processos do PSP

O PSP é composto por quatro níveis, numerados de 0 a 3. São eles: PSP0 (processo referencial), PSP1 (processo de planejamento pessoal), PSP2 (processo de gestão pessoal de qualidade) e por fim o PSP3 (processo pessoal cíclico). Esses processos serão detalhados a seguir (WILSON JÚNIOR,2000):

1. Processo referencial:

Práticas de medidas e alguns formatos de relatórios são estabelecidos e estes constituirão um referencial ou fundação sobre a qual será implantada a melhoria contínua do pessoal.

2. Processo de planejamento pessoal:

São introduzidos relatório de testes e práticas de estimativa de tamanho e recursos. Logo após, são adicionados o planejamento de tarefas e a elaboração de cronogramas.

3. Processo de gestão pessoal de qualidade

Introduz as técnicas de inspeção e revisão para auxiliar na detecção de defeitos o mais cedo possível, quando é menos custoso corrigi-los. Para isso, é feita a coleta e análise dos dados de defeitos de compilação e testes encontrados em programas anteriores. Assim, é possível criar listas de verificação mais ajustadas ao perfil de defeitos do programador.

4. Processo pessoal cíclico

O programa é dividido em módulos que possam ser tratados mais facilmente. A idéia é desenvolver o programa de forma interativa e incremental. Para isso, é de extrema importância que cada iteração tenha sua qualidade controlada, assumindo que a qualidade das anteriores está garantida. É preciso assegurar que a inclusão de novos módulos não afete a qualidade dos anteriores.

3.3.2.1 Vantagens do PSP

A utilização do PSP proporciona algumas vantagens, como por exemplo, o fato de que os desenvolvedores passarão a compreender melhor suas tarefas quando definem, medem e acompanham os trabalhos. Também passarão a usar uma estrutura de processos definida e critérios mensuráveis para que possam avaliar e aprender tanto com a própria experiência como com a dos outros. Com isso, poderão selecionar métodos e técnicas que se ajustam melhor aos tipos de tarefas que executam e às suas atividades. Outra vantagem é que os desenvolvedores tornar-se-ão membros mais produtivos ao fazerem uso de práticas bem definidas (CHIOSSI, CÔRTES, 2001).

Por fim, outra vantagem que se pode perceber é a diminuição de erros por linha de código, evitando o retrabalho.

3.3.3 MPS.BR

De acordo com o Ministério da Ciência e Tecnologia, em 2003 apenas trinta empresas produtoras de *software* tinha o certificado Sei/CMU de avaliações CMM, sendo que vinte e quatro dessas estavam no nível 2, cinco no nível 3, uma no nível 4 e nenhuma no nível 5. A SOFTEX (Programa Brasileiro de *Software* para Exportação), com o intuito de ajudar a solucionar esse problema, propôs um projeto denominado Melhoria de Processo do Software Brasileiro, o MPS-Br. (ARAÚJO et al, 2004)

A SOFTEX é responsável pela coordenação deste programa e conta com apoio do MCT (Ministério da Ciência e Tecnologia), da FINEP (Financiadora de Estudos e Projetos) e do BID (Banco Interamericano de Desenvolvimento). Duas estruturas apóiam o desenvolvimento das atividades do MPS.BR que são o Fórum de Credenciamento e Controle (FCC) e a Equipe Técnica do Modelo (ETM). Através delas o MPS.BR obtém a participação de representantes de Universidades, Instituições Governamentais, Centros de Pesquisa e de organizações privadas (SOFTEX, 2006).

O FCC é responsável por garantir que as Instituições Implementadoras (II) e Instituições Avaliadoras (IA) sejam submetidas a um processo adequado de credenciamento além de avaliar e atuar sobre o controle dos resultados obtidos pelo MPS.BR (SOFTEX, 2006).

A ETM atua sobre os aspectos técnicos relacionados ao Modelo de Referência (MR-MPS) e Método de Avaliação (MA-MPS), como por exemplo, a concepção e evolução do modelo, elaboração e atualização dos guias do MPS.BR, preparação de material e definição da forma de treinamento e de aplicação de provas, publicação de relatórios técnicos e interação com a comunidade visando a identificação e aplicação de melhores práticas (SOFTEX, 2006).

O projeto MPS.BR propõe definir um modelo de melhoria e avaliação de processo de *software*. Compreende duas metas: desenvolver e aprimorar o modelo MPS compatível com o CMMI e em conformidade com as normas ISO/IEC 12207 e 15504 e também implementar o modelo a um custo acessível em todas as regiões do país, com foco nas pequenas e médias empresas (ARAÚJO et al, 2004).

3.3.3.1 Descrição do MR - MPS

O MR-MPS (Modelo de Referência do MPS-Br) define sete níveis de maturidade que são uma combinação entre processos e sua capacidade. A definição de processo afirma o propósito e os resultados esperados de sua execução. Desta forma, permite avaliar e atribuir graus de efetividade na execução dos processos (SOFTEX, 2006).

Os níveis de maturidade estabelecem patamares de evolução de processos, caracterizando estágios de melhoria de implementação de processos na organização (ARAÚJO et al, 2004).

A capacidade do processo é caracterizada pela sua habilidade de alcançar os objetivos de negócio, tanto atuais quanto futuros. Está relacionada “com o atendimento aos atributos de processo associados aos processos de cada nível de maturidade” (SOFTEX, 2006).

Essa capacidade possui cinco atributos que são baseados na norma ISO/IEC 15504 e correspondem às práticas genéricas do CMMI. (ARAÚJO et al, 2004).

Esses atributos são:

- AP 1.1: O processo é executado.
- AP 2.1: O processo é gerenciado.
- AP 2.2: Os produtos de trabalho do processo são gerenciados
- AP 3.1: O processo é definido
- AP 3.2: O processo está implementado

É possível perceber através da tabela 1 que em cada nível existem processos que estão voltados para a qualidade. Isso acontece mesmo no nível mais baixo, com os processos de gerência de projeto e de requisitos. No nível F há um processo específico que visa garantir a qualidade. A partir do nível E existe a preocupação não só em garantir a qualidade dos processos, mas também na sua avaliação e melhoria.

Nível	Processos	Atributos do Processo
A (mais alto)	Implantação de Inovações na Organização	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Análise de Causas e Resolução	
B	Desempenho do Processo Organizacional	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência Quantitativa do Projeto	
C	Análise de Decisão e Resolução	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência de Riscos	
D	Desenvolvimento de Requisitos	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Solução Técnica	
	Integração do Produto	
	Verificação	
	Validação	
E	Treinamento	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Definição do Processo Organizacional	
	Avaliação e Melhoria do Processo	
	Adaptação do Processo para Gerência do Projeto	
F	Medição	AP 1.1, AP 2.1 e AP 2.2
	Gerência de Configuração	
	Aquisição	
	Garantia da Qualidade	
G	Gerência de Requisitos	AP 1.1 e AP 2.1
	Gerência do Projeto	

Tabela 1: Níveis de maturidade do MR-MPS
Fonte: Softex (2006).

3.3.3.2 Descrição dos Processos

1. Nível G – Parcialmente Gerenciado

O primeiro processo desse nível é o Gerência de Projeto (GPR). O GPR tem como objetivos identificar, estabelecer, coordenar e monitorar as atividades, tarefas e recursos que um projeto precisa para produzir um produto e/ou serviço, no contexto dos requisitos e restrições do projeto (SOFTEX, 2006).

Os principais resultados esperados são:

- O escopo do trabalho para o projeto está definido;
- As fases do ciclo de vida do projeto são definidas;

- Estuda-se a viabilidade de atingir as metas do projeto, considerando as restrições e os recursos disponíveis. Alguns ajustes podem ser realizados, se houver necessidade;
- As tarefas, os recursos e a infra-estrutura necessários para completar o trabalho são planejados;
- O cronograma e o orçamento do projeto são estabelecidos e mantidos;
- Os riscos do projeto são identificados e o seu impacto, probabilidade de ocorrência e prioridades de tratamento são determinados e documentados;
- Os recursos humanos para o projeto são planejados considerando o perfil e o conhecimento necessários para executá-lo;
- O planejamento do projeto é revisado com todos os interessados e o compromisso com o mesmo é obtido.

O segundo processo do nível G é o Gerência de Requisitos (GRE) que tem como propósito “gerenciar os requisitos dos produtos e componentes do produto do projeto e identificar inconsistências entre esses requisitos e os planos e produtos de trabalho do projeto” (SOFTEX, 2006).

Os principais resultados desejados são:

- Uma comunicação contínua com os fornecedores de requisitos é estabelecida e o entendimento dos requisitos é obtido;
- O comprometimento com os requisitos é estabelecido e mantido;
- Inconsistências entre os planos do projeto, os produtos de trabalho e os requisitos são identificadas e corrigidas;
- Mudanças nos requisitos são gerenciadas ao longo do projeto.

2. Nível F – Gerenciado

O primeiro processo do nível F é o Aquisição (AQU) que tem como propósito obter um produto e/ou serviço que satisfaça a necessidade do cliente (SOFTEX, 2006).

Os principais resultados esperados são:

- As necessidades de aquisição, as metas, os critérios de aceitação do produto e/ou serviço, os tipos e estratégia de aquisição são definidos;
- Os critérios de seleção do fornecedor são estabelecidos e usados para avaliar os potenciais fornecedores;

- O fornecedor é selecionado com base na avaliação das propostas e dos critérios estabelecidos;
- Um produto e/ou serviço que satisfaz a necessidade expressa pelo cliente é adquirido baseado na análise dos potenciais candidatos;
- A aquisição é monitorada de forma que as condições especificadas são atendidas, tais como: custo, cronograma e qualidade e, se necessário, ações corretivas são conduzidas;
- O produto e/ou serviço de software entregue é avaliado em relação ao acordado e os resultados da aceitação são documentados;
- O produto adquirido (caso pertinente) é incorporado ao projeto.

O segundo processo é o Gerência de Configuração (GCO). O propósito é "estabelecer e manter a integridade de todos os produtos de trabalho de um processo ou projeto e disponibilizá-los a todos os envolvidos" (SOFTEX, 2006).

Os seguintes resultados são esperados:

- Os itens de configuração são identificados;
- Os itens de configuração gerados pelo projeto são definidos e colocados sob uma linha base;
- É estabelecido e mantido um Sistema de Gerência de Configuração;
- As modificações e liberações dos itens de configuração são controladas;
- As modificações e liberações são disponibilizadas para todos os envolvidos;
- A situação dos itens de configuração e as solicitações de mudanças são registradas, relatadas e o seu impacto é analisado;
- A completeza e a consistência dos itens de configuração são asseguradas;
- O armazenamento, o manuseio e a entrega dos produtos de trabalho são controlados;
- É estabelecida a integridade das linhas bases e mantida através de auditoria da configuração e de registros da Gerência de Configuração.

O terceiro processo é chamado de Garantia da Qualidade (GQA). Ele tem como objetivo garantir que os produtos de trabalho e a execução dos processos estejam de acordo com os planos e recursos predefinidos (SOFTEX, 2006).

Os principais resultados que se espera desse processo são:

- A aderência dos produtos aos padrões, procedimentos e requisitos aplicáveis é avaliada objetivamente;
- Os produtos de trabalho são avaliados antes de serem entregues ao cliente e em marcos predefinidos ao longo do ciclo de vida do projeto;
- Os problemas e as não-conformidades são identificados, registrados e comunicados;
- O escalonamento das ações corretivas para níveis superiores é realizado, quando necessário, de forma a garantir a solução das mesmas;
- A aderência ao processo de Garantia da Qualidade e de seus produtos de trabalho aos padrões, procedimentos e requisitos aplicáveis é avaliada objetivamente.

O último processo desse nível é o de Medição (MED). Ele objetiva coletar e analisar os dados relativos aos produtos desenvolvidos e aos processos implementados na organização e em seus projetos, de forma a apoiar os objetivos organizacionais (SOFTEX, 2006).

Espera-se os seguintes resultados:

- Objetivos e atividades de medição são estabelecidos a partir das necessidades de informação e objetivos da organização;
- Um conjunto adequado de medidas, orientado pelas necessidades de informação e objetivos de medição, é identificado e/ou desenvolvido, priorizado, documentado, revisado e atualizado;
- As atividades coleta e armazenamento são especificadas, incluindo-se métodos e ferramentas;
- As atividades de análise são especificadas, incluindo-se métodos e ferramentas;
- Os dados requeridos são coletados e analisados;
- Os dados e os resultados são armazenados;
- As informações produzidas são usadas para apoiar decisões e para fornecer uma base objetiva para comunicação aos interessados.

3. Nível E – Parcialmente definido

Esse nível possui quatro processos. O primeiro deles é o Adaptação do Processo para Gerência do Projeto (APG) que visa estabelecer e gerenciar o projeto e envolver os interessados de acordo com o processo definido e integrado que é adaptado do conjunto de processos-padrão da organização (SOFTEX, 2006).

Os principais resultados esperados desse processo são:

- Um processo definido para o projeto é estabelecido de acordo com a estratégia para adaptação de processo da organização. ;
- O planejamento e as estimativas das atividades do projeto são feitos baseados no repositório de estimativas e no conjunto de ativos de processos organizacionais;
- O Plano do Projeto e outros planos que afetam o projeto são integrados;
- Os produtos de trabalho, medições e experiências documentadas contribuem para os ativos de processos organizacionais;
- Dependências críticas são identificadas, negociadas e acompanhadas com os interessados;
- Questões pertinentes são resolvidas com os interessados.

O segundo processo é a Avaliação e Melhoria do Processo Organizacional (AMP). Seu propósito é determinar o quanto os processos-padrão da organização contribuem para planejar e implementar melhorias contínuas nos processos com base no entendimento de seus pontos fortes e fracos (SOFTEX, 2006).

Os principais resultados esperados são:

- A descrição das necessidades e os objetivos dos processos da organização estão estabelecidos e mantidos;
- As informações e os dados relacionados ao uso dos processos-padrão para projetos específicos existem e são mantidos em repositório específico;
- Revisões dos processos-padrão da organização são realizadas em intervalos adequados para assegurar sua adequação e efetividade e manter o entendimento de seus pontos fortes e fracos;
- A implantação de ativos do processo organizacional ou de alterações nestes ativos são implementadas de maneira controlada na organização;

- Experiências relacionadas aos processos são incorporadas nos ativos do processo organizacional;
- Dados técnicos, históricos e resultantes das avaliações são analisados e usados para melhorar os processos, recomendar mudanças nos projetos e determinar necessidades de mudanças tecnológicas.

O terceiro processo é o de Definição do Processo Organizacional (DFP). Ele tem o objetivo de estabelecer e manter um conjunto de ativos dos processos organizacionais aplicável às necessidades de negócio da organização (SOFTEX, 2006).

Os principais resultados são:

- Um conjunto de processos-padrão da organização é definido e documentado, juntamente com a indicação da aplicabilidade de cada processo;
- O conjunto de processos definido é mantido em uma biblioteca de ativos de processos da organização com mecanismos de consulta e recuperação;
- Tarefas, atividades e produtos de trabalho associados aos processos-padrão são identificados e detalhados, juntamente com as características de desempenho esperadas;
- São desenvolvidas estratégias para adaptação do processo-padrão de acordo com as necessidades dos projetos;
- O repositório de medidas da organização é estabelecido e mantido.

O último processo é o de Treinamento (TRE). O propósito desse processo é prover a organização e os projetos com profissionais que possuam os conhecimentos e as habilidades necessárias para executar suas funções de forma efetiva (SOFTEX, 2006).

Os resultados esperados nesse processo são:

- Uma estratégia de treinamento é planejada e implementada com o objetivo de atender as necessidades de treinamento dos projetos e da organização;
- As necessidades de treinamento que são responsabilidade da organização são identificadas;

- Para garantir que todos os indivíduos tenham o conhecimento e habilidades requeridas para executar suas atividades, é estabelecida uma estratégia para treinamento que contemple mecanismos, materiais e instrutores qualificados;
- O treinamento é conduzido e registrado;
- A efetividade do treinamento é avaliada.

4. Nível D – Largamente definido

Esse nível é composto por cinco processos, a começar pelo Desenvolvimento de Requisitos (DRE). Ele visa estabelecer os requisitos dos componentes do produto, do produto e do cliente (SOFTEX, 2006).

Seus principais resultados obtidos são:

- As necessidades e expectativas, restrições e requisitos de interface do cliente são identificadas;
- Um conjunto definido de requisitos funcionais e não-funcionais que descrevem a solução do problema a ser resolvido é estabelecido a partir das necessidades, expectativas, restrições e requisitos do cliente e da interface;
- Conceitos operacionais e cenários são desenvolvidos;
- Os requisitos são analisados para assegurar que são necessários e suficientes e para balancear as necessidades dos interessados com as restrições existentes;
- Os requisitos são validados.

O segundo processo é Integração do Produto (ITP) que tem como propósito compor os componentes do produto, produzindo um produto integrado consistente com o projeto e demonstrar que os requisitos funcionais e não-funcionais são satisfeitos para o ambiente alvo ou equivalente (SOFTEX, 2006).

Os principais resultados esperados são:

- Um ambiente para integração dos componentes do produto é estabelecido e mantido;
- A compatibilidade das interfaces internas e externas dos componentes do produto é assegurada;

- As definições, projeto e mudanças nas interfaces internas e externas são gerenciados para os componentes dos produtos e produtos;
- Cada componente do produto é verificado, utilizando-se critérios definidos, para confirmar que os mesmos estão prontos para a integração;
- Os componentes do produto integrados são avaliados e os resultados da integração são registrados;
- Uma estratégia de regressão é desenvolvida e aplicada para uma nova verificação do produto quando ocorre uma mudança nos componentes do produto (incluindo requisitos, projeto e códigos associados);
- O produto e a documentação relacionada são preparados e entregues ao cliente.

O terceiro processo desse nível é o de Solução Técnica (STE). Ele objetiva projetar, desenvolver e implementar soluções para atender aos requisitos (SOFTEX, 2006).

Os resultados desejados são:

- Alternativas de solução para atender aos requisitos definidos são desenvolvidas de acordo com critérios identificados;
- Soluções são selecionadas para o produto ou componentes do produto com base em cenários definidos e em critérios identificados;
- O produto ou componente do produto é projetado e documentado;
- As interfaces entre os componentes do produto são projetadas com base em critérios predefinidos;
- Uma análise dos componentes do produto é conduzida para decidir sobre sua construção, compra ou reuso;
- Os componentes do produto e a sua documentação associada são implementados e verificados de acordo com o projeto;
- A documentação é identificada, desenvolvida e disponibilizada de acordo com os padrões identificados;
- A documentação é mantida de acordo com os critérios definidos.

O penúltimo processo desse nível é o de Validação (VAL). Ele tem o propósito de confirmar que um produto ou componente do produto atenderá a seu

uso pretendido quando colocado no ambiente para o qual foi desenvolvido (SOFTEX, 2006).

Os resultados esperados são:

- São identificados os produtos de trabalho a serem validados;
- Uma estratégia de validação é desenvolvida estabelecendo cronograma, participantes envolvidos, métodos para validação e qualquer material a ser utilizado na validação;
- Critérios para validação dos produtos de trabalho a serem validados são identificados e um ambiente para validação é estabelecido;
- Atividades de validação são executadas para garantir que os produtos de *software* estão prontos para uso no ambiente operacional pretendido;
- Evidências de que os produtos de software desenvolvidos estão prontos para o uso pretendido são fornecidas;
- Resultados de atividades de validação são analisados e disponibilizados para os interessados.

O último processo é o de Verificação (VER) que objetiva confirmar que cada serviço e/ou produto de trabalho do processo ou do projeto reflete apropriadamente os requisitos especificados (SOFTEX, 2006).

Os resultados esperados são:

- Produtos de trabalho a serem verificados são identificados;
- Uma estratégia de verificação é desenvolvida e implementada estabelecendo cronograma, revisores envolvidos, métodos para verificação e qualquer material a ser utilizado na verificação;
- Critérios para verificação dos produtos de trabalho a serem verificados são identificados e um ambiente para verificação é estabelecido;
- Atividades de verificação, incluindo testes e revisões por pares, são executadas;
- Defeitos são identificados, registrados e ações corretivas são realizadas.
- Resultados de atividades de verificação são analisados e disponibilizados para os interessados.

5. Nível C – Definido

É composto apenas por dois processos. O primeiro é o deAnálise de Decisão e Resolução (ADR). Ele propõe analisar possíveis decisões usando um processo formal da avaliação das alternativas identificadas em relação a critérios estabelecidos (SOFTEX, 2006).

Os principais resultados que são esperados:

- O problema ou questão a ser objeto de um processo formal de tomada de decisão é definido;
- Guias para a análise de decisão são estabelecidos e mantidos;
- Alternativas de solução aceitáveis para o problema ou questão são identificadas;
- Soluções alternativas são avaliadas usando os critérios e métodos estabelecidos;
- Decisões são baseadas na avaliação das alternativas utilizando os critérios de avaliação estabelecidos.

O outro processo é o de Gerência de Riscos (GRI). O propósito é identificar, gerenciar e reduzir continuamente os riscos em nível organizacional e de projeto (SOFTEX, 2006).

Alguns resultados desejados são:

- O escopo da gerência de riscos é determinado;
- As origens e as categorias de riscos são determinadas, os parâmetros usados para quantificação da probabilidade e severidade são definidos e as ameaças e suas fronteiras para cada categoria de risco são definidas;
- Estratégias apropriadas para a gerência de riscos são definidas e implementadas;
- Os riscos do projeto são identificados e documentados incluindo seu contexto, condições e possíveis conseqüências para o projeto e as partes que serão afetadas;
- Os riscos são priorizados, estimados e classificados de acordo com as categorias e os parâmetros definidos;
- Os riscos são analisados e a prioridade de aplicação dos recursos para o monitoramento desses riscos é determinada;

- As medições de desempenho nas atividades de tratamento de risco são coletadas;
- Ações apropriadas são executadas para corrigir ou evitar o impacto dos riscos.

6. Nível B – Gerenciado Quantitativamente

É o penúltimo nível e também é composto por dois processos. O primeiro é o de Desempenho do Processo Organizacional (DEP). Ele visa estabelecer e manter um entendimento quantitativo do desempenho dos processos-padrão da organização para apoiar os objetivos para qualidade e para o desempenho dos processos. Visa também fornecer dados, linhas-básicas e modelos para gerenciar quantitativamente os projetos da organização (SOFTEX, 2006).

Os resultados desejados são:

- A partir do conjunto de processos-padrão da organização, são selecionados os processos e/ou elementos de processos que serão objeto de análise de desempenho;
- Medidas para análise do desempenho dos processos da organização são estabelecidas e mantidas;
- Objetivos quantitativos para qualidade e para o desempenho dos processos da organização são definidos e mantidos;
- Linhas bases de desempenho dos processos da organização são estabelecidas;
- Modelos de desempenho do processo para o conjunto de processos-padrão da organização são estabelecidos e mantidos.

O segundo processo desse nível é o de Gerência Quantitativa do Projeto (GQP). Ele propõe gerenciar quantitativamente o processo definido para o projeto de forma a alcançar os objetivos para qualidade e para o desempenho do processo estabelecido para o projeto (SOFTEX, 2006).

Alguns resultados esperados são:

- Os objetivos para qualidade e para o desempenho do processo para o projeto são estabelecidos e mantidos;

- Os sub-processos mais adequados para compor o processo definido para o projeto são selecionados com base na estabilidade histórica, em dados de capacidade e em critérios previamente estabelecidos;
- O projeto é monitorado para determinar se seus objetivos para qualidade e para o desempenho do processo serão atingidos e ações corretivas são identificadas quando necessário;
- Medidas e técnicas de análise para gerenciar estatisticamente os sub-processos escolhidos são selecionadas;
- O desempenho dos sub-processos escolhidos para gerência quantitativa é monitorado para determinar a sua capacidade de satisfazer os seus objetivos para qualidade e para o desempenho e ações são identificadas quando for necessário tratar deficiências dos sub-processos;
- Dados estatísticos e de gerência da qualidade são incorporados ao repositório de medidas da organização.

7. Nível A – Em otimização

Por fim, esse é o nível mais alto do MPS-Br e também possui dois processos. O primeiro é o de Implantação de Inovações na Organização (IIO). Ele tem como objetivo selecionar e implantar melhorias incrementais e inovadoras que melhorem os processos e as tecnologias da organização. Essas melhorias apóiam os objetivos de qualidade e de desempenho dos processos da organização, que são derivados de seus objetivos de negócio (SOFTEX, 2006).

Os principais resultados a serem alcançados são:

- Propostas de melhoria do processo e de melhoria tecnológica são coletadas e analisadas;
- As melhorias inovadoras que aumentem a qualidade da organização e o desempenho do seu processo são identificadas e analisadas;
- Projetos-piloto são planejados e realizados para identificar as melhorias no processo e as melhorias tecnológicas que serão implementadas;
- As propostas de melhoria no processo e de melhorias tecnológicas são selecionadas com base em critérios quantificáveis definidos a partir dos objetivos de qualidade e de desempenho do processo da organização;

- A implantação das melhorias no processo e das melhorias tecnológicas é realizada e gerenciada a partir dos planos;
- Os efeitos resultantes da implantação das melhorias no processo e das melhorias tecnológicas são medidos e documentados.

O segundo processo do último nível é o de Análise de Causas e Resolução (ARC). Ele tem o propósito de identificar causas de defeitos e de outros problemas e tomar ações para prevenir suas ocorrências no futuro (SOFTEX, 2006).

Os resultados desejados são:

- Dados de defeitos e de outros problemas são selecionados para análise;
- A análise da causa de defeitos e de outros problemas selecionados é realizada com as pessoas responsáveis por realizar a tarefa para identificar a sua raiz;
- Ações necessárias para evitar a ocorrência futura de defeitos e outros problemas similares aos selecionados são propostas e documentadas;
- Ações propostas e documentadas que foram desenvolvidas durante a análise da causa e que possuem uma relação custo/benefício satisfatória são implementadas para remover as causas dos defeitos e dos problemas analisados e evitar ocorrências futuras;
- O efeito das mudanças no desempenho dos processos é medido e avaliado para obter evidência de que as mudanças nos processos corrigiram o problema e melhoraram o desempenho;
- Dados da análise de causas e resolução são armazenados para que outros projetos e organizações possam realizar mudanças apropriadas nos processos e alcançar resultados similares.

Conforme foi visto até aqui com as metodologias de desenvolvimento e os métodos que avaliam a qualidade, os processos das empresas produtoras de *software* devem ser bem definidos, acompanhados e avaliados de forma sistemática para que o projeto final seja satisfatório. No próximo capítulo serão analisados, sob o ponto de vista da qualidade, os processos de desenvolvimento das empresas locais produtoras de sistemas.

4. MAPEAMENTO DOS PROCESSOS DE DESENVOLVIMENTO EM EMPRESAS PRODUTORAS DE *SOFTWARE* EM VITÓRIA DA CONQUISTA: UMA ANÁLISE DA QUALIDADE

O objetivo dessa pesquisa foi colher informações sobre os processos de desenvolvimento de *software* utilizados pelas empresas de Vitória da Conquista. Foi elaborado um questionário com perguntas que abordaram vários assuntos, com o propósito de coletar dados sobre os perfis das empresas, a equipe e as fases de desenvolvimento, uso de ferramentas para realização de algumas atividades, metodologia e documentação das atividades, cumprimento de prazos e custos e finalmente sobre padrões de qualidade.

O questionário foi aplicado em algumas empresas produtoras de *software* na cidade de Vitória da Conquista, que formaram um total de oito. Os nomes das empresas se encontram no apêndice B desse documento.

Abaixo segue uma análise detalhada dos resultados obtidos. Os assuntos estão separados em seções para melhor facilitar o entendimento. Os dados são mostrados através de gráficos, onde em cada um deles o percentual correto de cada resposta é mostrado. Como muitas perguntas tinham mais de uma opção de resposta, o número de respostas passou a ser maior do que o de empresas, assim, a soma dos valores dos seus respectivos gráficos não resulta cem por cento. No apêndice A segue o modelo do questionário aplicado nas empresas.

4.1 PERFIS DAS EMPRESAS

Vive-se na sociedade de informação e conhecimento. A tecnologia da informação é a tônica atual. Em Vitória da Conquista, o cenário é o mesmo, pois se percebe uma tendência no crescimento com relação às empresas produtoras de *software*.

Um total de sessenta e três por cento das empresas entrevistadas atua no mercado há mais de quatro anos. Porém, pode-se perceber através do gráfico 1, que nesse último ano tem crescido o número de empresas produtoras de *software* nessa cidade, totalizando vinte e cinco por cento.

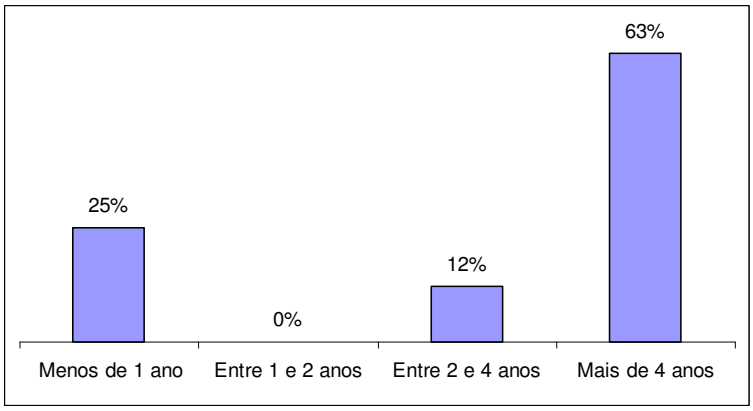

Gráfico 1: Tempo de atuação das empresas produtoras de *software*

Referente aos campos de atuação dessas empresas, muitas delas estão presentes em mais de um deles e, conseqüentemente, em mais de um segmento também. A maioria (oitenta e sete e meio por cento) está atuando na prestação de serviços. Outro ramo forte é o do comércio, pois quase a metade (trinta e sete e meio por cento) das empresas entrevistadas atua nele. Os principais segmentos são as áreas de tecnologia da informação, saúde, calçados e confecções, conforme pode ser observado nos gráficos 2 e 3.

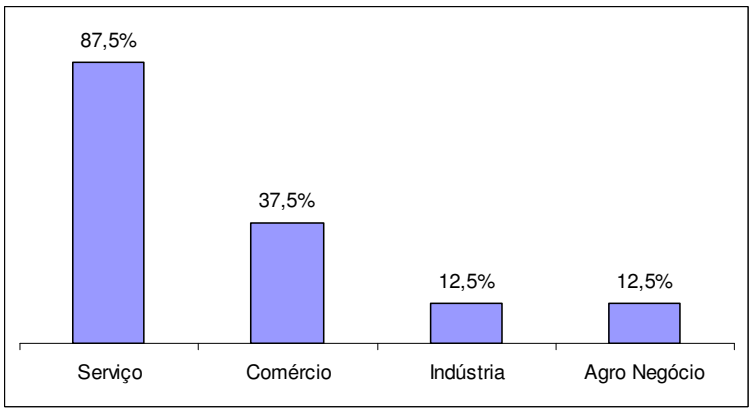

Gráfico 2: Ramo de atuação das empresas produtoras de *software*

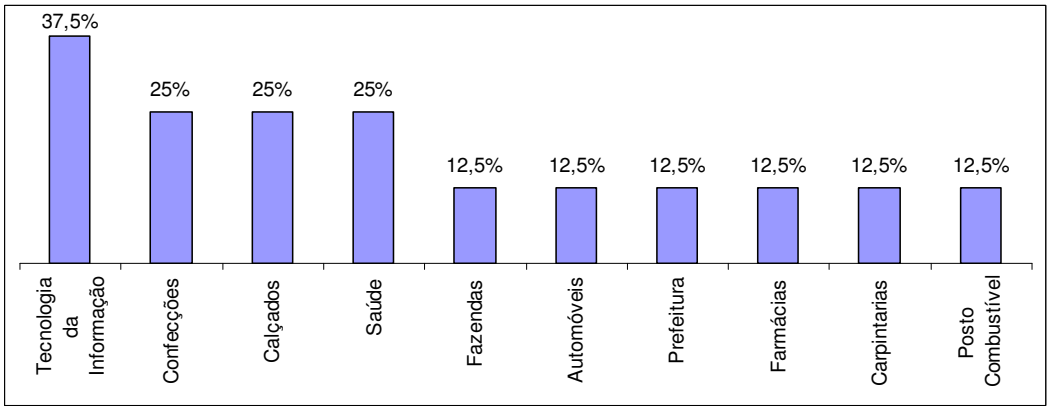

Gráfico 3: Segmento de atuação das empresas produtoras de *software*

Analisando os dados pode-se perceber que, apesar de se encaixarem no perfil de pequenas ou micro empresas, há uma preocupação em fazer investimentos para obter alguma melhoria na produção de sistemas. Maior parte dos investimentos são hardware (oitenta e sete e meio por cento), marketing (setenta e cinco por cento), *software* e treinamento da equipe (sessenta e dois e meio por cento).

Foi possível perceber que muitas dessas empresas não se preocupam em investir em certificação, o que pode ser uma conseqüência do alto custo para se obter uma certificação do porte do ISO e CMMI, que são os padrões mais conhecidos dessas empresas. Porém, o padrão mais indicado seria o MPS-Br, contudo é pouco conhecido pelas empresas, como será visto na seção 4.7 desse capítulo. O gráfico 4 ilustra esses dados.

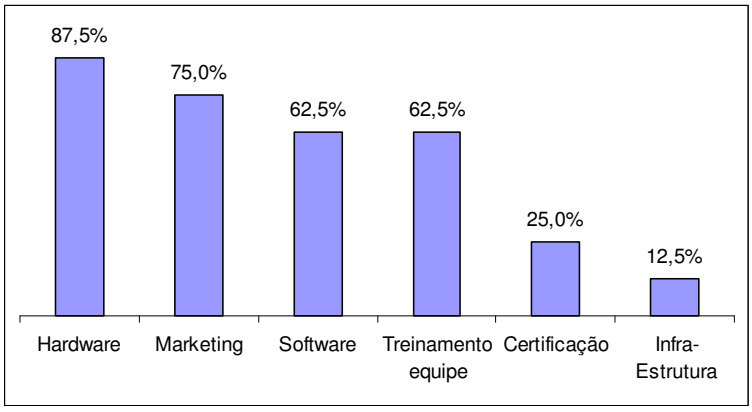

Gráfico 4: Investimentos

4.2 EQUIPE DE DESENVOLVIMENTO

A equipe de desenvolvimento de sistemas é chave para o sucesso de projetos. É importante que ela esteja bem preparada e que exista um número suficiente de colaboradores para que não haja sobrecarga de atividades e assim o projeto possa prosseguir sem problemas.

Apenas uma empresa das entrevistadas terceiriza seus serviços, mas é importante ressaltar que os serviços são prestados por mão de obra local. Porém, o tamanho das equipes ainda é relativamente pequeno, sendo que um pouco mais da metade (cinqüenta e sete por cento) das empresas tem uma equipe entre duas e quatro pessoas, como pode ser observado no gráfico 5.

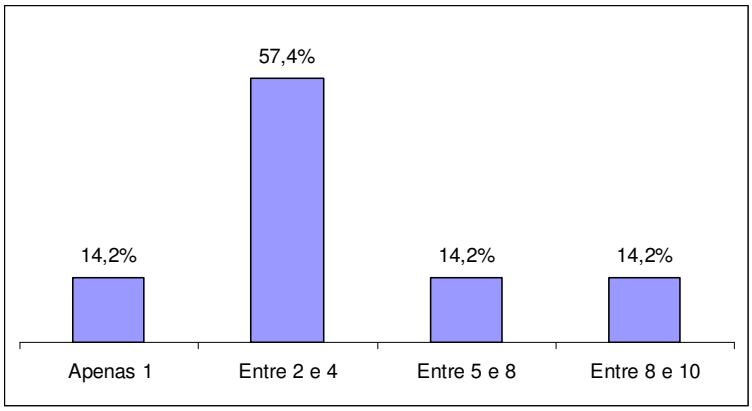

Gráfico 5: Tamanho da equipe de desenvolvimento

Há uma preocupação por parte das empresas em dar oportunidades àquelas pessoas que estão fazendo um curso de nível superior e até mesmo alguma especialização. Através dos gráficos 6 e 7, pode-se notar que em todas as empresas existem estudantes do curso de Ciência da Computação, e em quatorze por cento delas existem pessoas dos cursos Engenharia da Computação e Administração. Logo espera-se que comece a haver uma sistematização das tarefas e procedimentos, pois as pessoas foram formadas em um ambiente acadêmico contribuindo com a formalização do profissional da área junto ao mercado local.

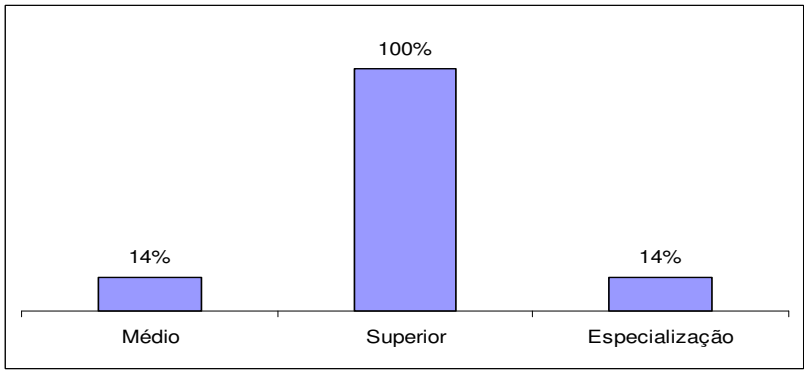

Gráfico 6: Escolaridade das equipes das empresas produtoras de *software*

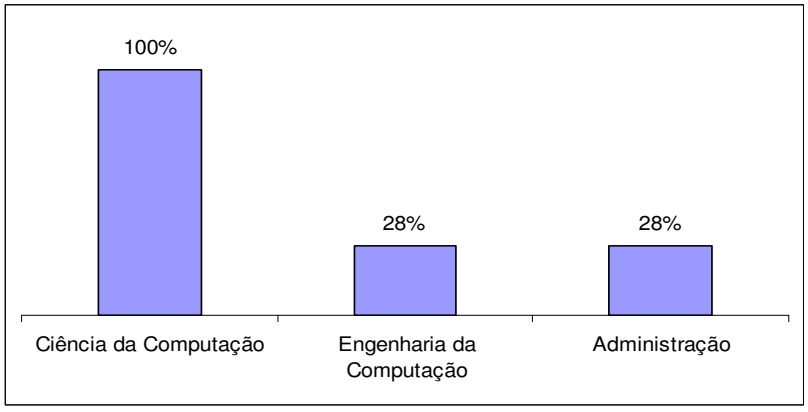

Gráfico 7: Cursos superiores realizados pelas equipes

4.3 ATIVIDADES DE DESENVOLVIMENTO

O processo de desenvolvimento de sistemas é composto por cinco etapas: concepção do projeto, planejamento e gerenciamento, desenvolvimento, testes, validação e manutenção do *software* criado. Cada fase possui algumas atividades inerentes a ela, e cada tarefa realizada corretamente contribuirá para o sucesso final de todo o projeto. Por isso, cada uma dessas atividades deve ser realizada por um profissional que tenha o perfil adequado e principalmente devem ser documentadas, para que as informações do projeto sejam acessíveis para toda a equipe e não apenas às pessoas que a realizam. (SCOTT, 2003).

A primeira fase é a de concepção, na qual o *software* é definido com seus requisitos e funcionalidades. Nesta fase, as necessidades do cliente são coletadas e as características do novo sistema são identificadas.

Essas duas atividades são de extrema importância, pois são baseadas nas informações coletadas que todo o sistema será construído. Um levantamento superficial dos requisitos pode comprometer todo o projeto, pois uma funcionalidade mal entendida terá que ser corrigida, prejudicando os compromissos de prazo e custos estabelecidos.

Na empresas locais, essas atividades estão sendo realizadas por um analista de negócios, qualidade e sistemas, conforme observado nos gráficos 8 e 9. Um analista de sistemas trabalhando em conjunto com um de negócios para realização dessa tarefa seria ideal. O analista de sistemas poderá identificar melhor os requisitos enquanto que o de negócios cuidaria do contrato, estabelecendo custos e prazos de entrega. Para um bom gerenciamento de projeto, é essencial que essas duas atividades sejam documentadas para que as informações estejam acessíveis.

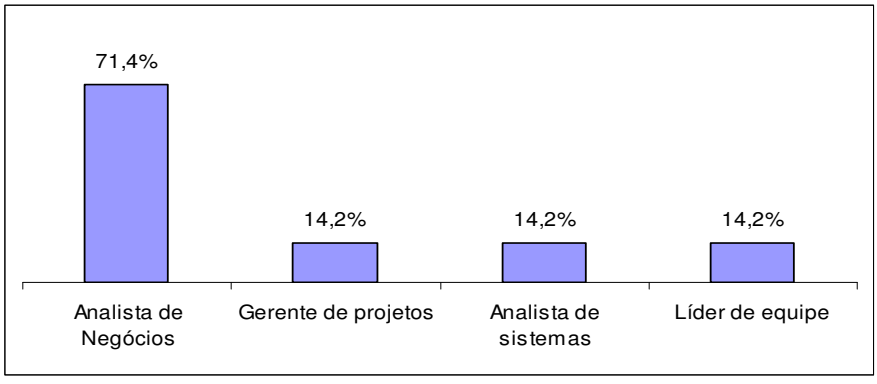

Gráfico 8: Responsáveis por levantar necessidades do cliente

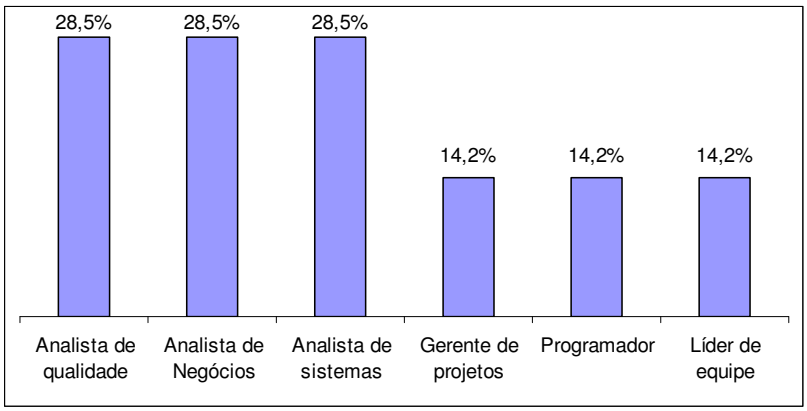

Gráfico 9: Responsáveis por identificar requisitos

Para uma empresa se enquadrar pelo menos no nível G do MPS-Br, o gerenciamento de requisitos deve ser feito de forma sistemática. É preciso uma comunicação contínua com os fornecedores de requisitos e a mudança deles deve ser gerenciada e devidamente documentada ao longo do projeto. Porém, a pesquisa revelou que apenas quatorze por cento das empresas entrevistadas fazem a documentação das reuniões com o cliente. Isso certamente gerará transtornos mais adiante, como por exemplo, a alteração de requisitos. O gráfico 10 mostra que as empresas apontam o cliente como responsável para mudança de requisitos, seja pelo fato de que ele não soube explicar (com cinqüenta e sete por cento) ou porque mudou ao longo do projeto (com quarenta e dois por cento).

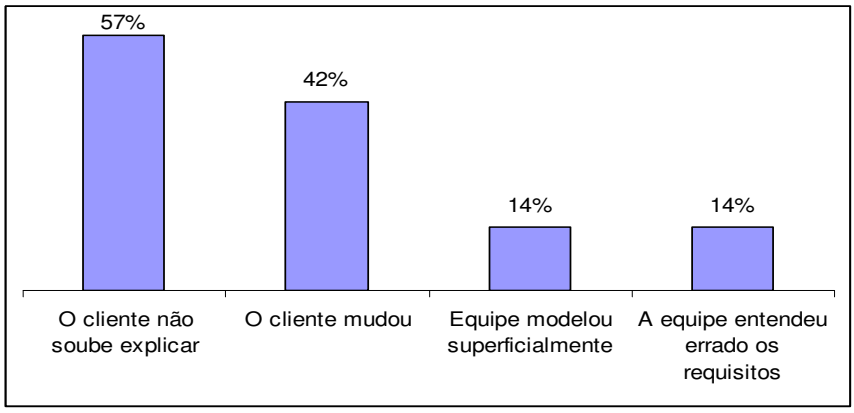

Gráfico 10: Razões da mudança de Requisitos

A segunda etapa é denominada planejamento e gerenciamento do projeto Nela, as tarefas a serem realizadas são: elaborar o plano, acompanhar e gerenciar o projeto, identificar os riscos, fazer a documentação das atividades e validar o projeto. A partir do nível G do MPS-Br já existe a preocupação em realizar corretamente

essas tarefas. Porém, oitenta e cinco por cento das empresas não se preocupam em gerenciar os riscos, resolvendo os problemas à medida que vão surgindo.

Em cinqüenta e sete por cento das empresas, o planejamento é elaborado pelo gerente de projetos, o que é o ideal, pois como o próprio nome sugere, ele é responsável pelo gerenciamento do projeto. O gráfico 11 mostra os outros perfis que também executam essas tarefas, mas não deveriam. Outro problema é que somente vinte e oito e meio por cento das empresas fazem a documentação do planejamento. Sem a documentação dessas atividades, não é possível caracterizar as empresas locais nem no nível G do MPS-Br.

Sem esse documento, o gerenciamento do projeto fica comprometido, pois nem o gerente terá conhecimento sobre os prazos de entrega e responsáveis pela realização das atividades, não havendo assim o efetivo controle e acompanhamento que precisam existir para conclusão do projeto.

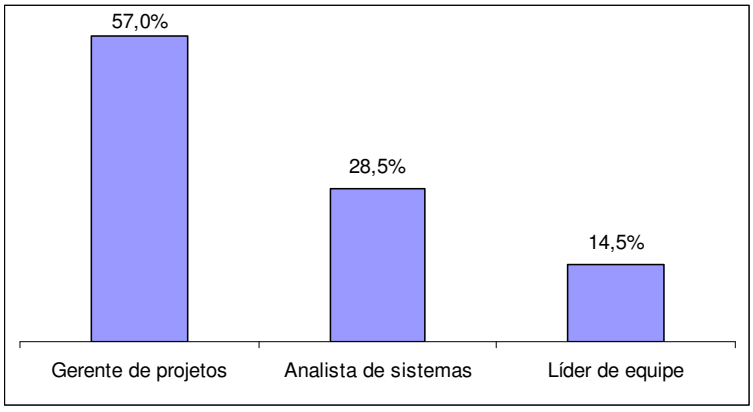

Gráfico 11: Responsáveis pela elaboração do plano de projeto

Com o projeto em mãos, é preciso validá-lo junto ao cliente em momentos como início, final, mas também existe o compromisso de estreitar a relação no sentido de se fazer entregas parciais com os respectivos artefatos, seja software ou documentação. É feito um acordo sobre os compromissos de prazos, custos, funcionalidades e necessidades em geral. Seria ideal que essa tarefa fosse realizada pelo gerente de projetos, pois ele foi o responsável por elaborar o plano, após discussão com a equipe sobre as questões de implementação. O gráfico 12 mostra que em apenas vinte e oito e meio por cento das empresas quem executa essa atividade é o gerente de projetos, coincidindo com o analista de sistemas e o programador.

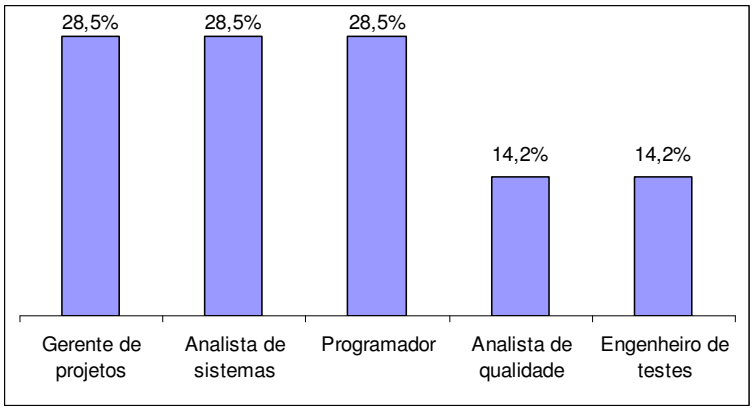

Gráfico 12: Responsáveis pela validação do projeto

Durante o andamento do projeto, é necessário fazer o gerenciamento do mesmo. É preciso fazer um acompanhamento das atividades, ou seja, saber se o projeto está sendo realizado dentro do previsto. Caso não esteja, é preciso tomar decisões para que os problemas surgidos sejam corrigidos ou diminuir o seu impacto, caso a correção não seja possível. Assim, é natural que essa tarefa também seja realizada pelo gerente de projetos, pois ele foi o responsável pelo planejamento e validação com o cliente, assim, precisa saber sobre tudo o que está acontecendo. Em cinqüenta e sete por cento das empresas, essa tarefa está sendo realizada por esse perfil, como mostra o gráfico 13. A dúvida é se outros profissionais estão acompanhando efetivamente o projeto.

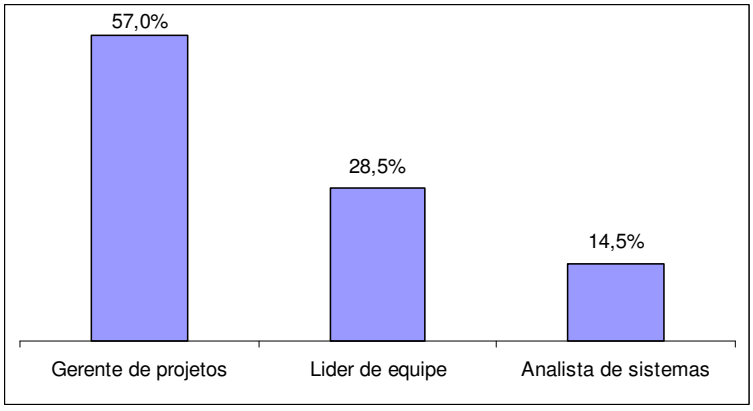

Gráfico 13: Responsáveis pelo gerenciamento do projeto

Para que o gerenciamento prossiga com mais facilidade, é preciso que as atividades sejam todas documentadas, desde o início até o fim do projeto. Caso não haja documentação, qualquer erro que apareça será tratado com dificuldade, pois não se saberá quem é o responsável e nem as suas conseqüências. Se acontecer a saída de algum membro da equipe, ele levará junto o conhecimento do que foi feito e o novo integrante terá dificuldade em ter informações do projeto.

Porém, sem a documentação do planejamento, como foi visto anteriormente, o gerenciamento está seriamente comprometido. A pesquisa revelou também que apenas algumas pessoas da equipe documentam suas atividades, sendo que cinqüenta e sete por cento são os programadores. O ideal seria que toda a equipe a fizesse, o que acontece em apenas quatorze por cento das empresas. Para que as empresas se encaixem no nível G do MPS-Br, as documentações de todas as atividades devem ser feitas e atualizadas sistematicamente, o que não acontece nas empresas locais. O gráfico 14 mostra esses dados.

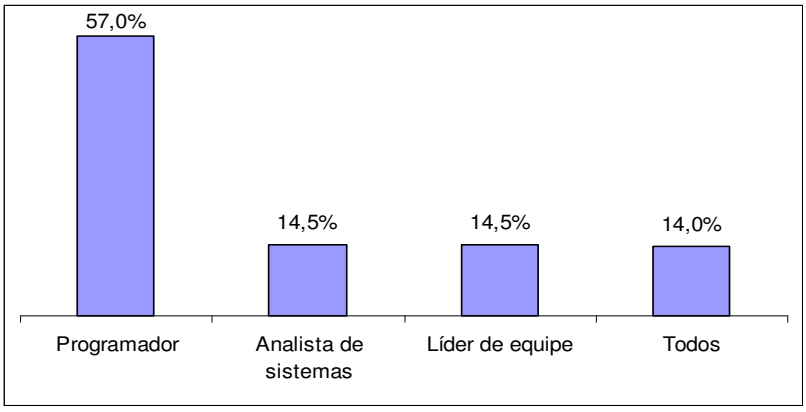

Gráfico 14: Responsáveis pela documentação das atividades

A terceira etapa se refere ao desenvolvimento do sistema, na qual o software será construído. É um das fases mais demoradas, é composta pela especificação e implementação das funcionalidades.

Especificar as funcionalidades é uma tarefa que dever ser realizada com muito cuidado. Um simples erro pode gerar sérios transtornos como, por exemplo, implementações desnecessárias de funcionalidades que não são requisitos, comprometendo o prazo de entrega do sistema.

Foi possível concluir que essa atividade está seriamente comprometida. Quatorze por cento das empresas entrevistadas afirmaram não realizar essa tarefa, o que está incorreto, devido à sua importância. Ainda assim, das empresas que realizam, apenas cinqüenta e sete por cento fazem a respectiva documentação. Assim, as empresas locais não poderiam se enquadrar no nível G do MPS-Br. Um fato positivo é que, em metade das empresas, essa atividade é realizada pelo analista de sistemas, o que é o mais adequado, pois ele foi o responsável por identificar os requisitos. O gráfico 15 mostra esses dados.

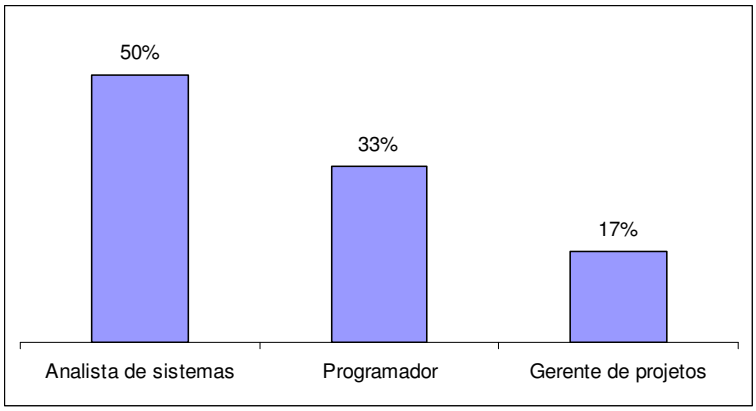

Gráfico 15: Responsáveis pela especificação das funcionalidades

Após especificação das funcionalidades, chegou a vez da implementação. É a tarefa mais demorada de todo o projeto, pois todo o sistema será construído. Porém, o sucesso da implementação depende da modelagem bem feita e levantamento dos requisitos. Em todas as empresas, os responsáveis pela implementação são os programadores, o que já era esperado. O que falta ser feito nessa etapa, mais uma vez é a sua documentação, para que o projeto não seja comprometido com a saída de algum membro. Somente quarenta e dois por cento das empresas realizam a documentação dessa fase, impedindo assim que consigam uma certificação nível G do MPS-Br.

A quarta etapa é a de testes e validação do *software* em questão. A fase de testes é essencial, pois ela aponta erros que passaram sem ser notados até então, e cria-se a oportunidade de corrigi-los antes que o sistema seja entregue. Seria interessante que acontecesse em paralelo à implementação, ou seja, à medida que os módulos fossem criados já seriam logo testados antes da integração com todo o sistema. Essa fase é constituída pelas seguintes atividades: planejar, implementar e executar os testes. Inicialmente somente na empresa e depois junto com o cliente.

Porém, foi possível perceber que as empresas não dão a devida atenção a essa fase, deixando de realizar alguma dessas tarefas. E quando são executadas, apenas quatorze por cento fazem as respectivas documentações. Desta forma, as empresas locais não estão de acordo com os propósitos do nível G do MPS-Br.

Vinte e oito e meio por cento das empresas não elaboram os planos de teste. Isso significa que nessas empresas os testes realizados acontecem de forma desorganizada. Das empresas que elaboram, em sessenta por cento delas o responsável por essa tarefa é o engenheiro de testes, que é o natural. Outros perfis são mostrados no gráfico 16.

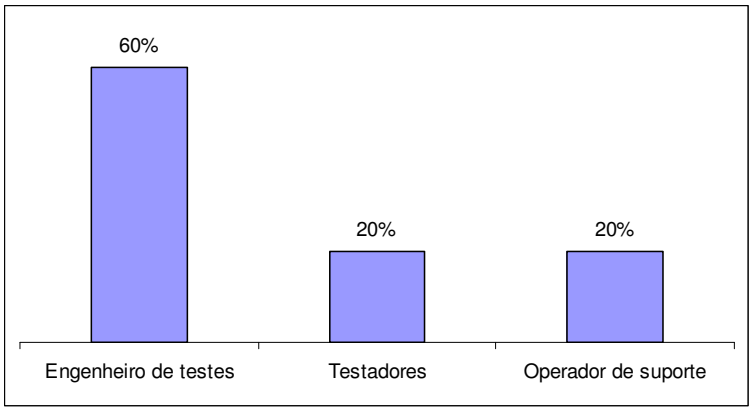

Gráfico 16: Responsáveis por elaborar plano de testes

Após o planejamento, é realizada a implementação dos testes. Mais uma vez, algumas empresas não executam essa atividade, totalizando quatorze e meio por cento. Das empresas que implementam, em metade o programador é o responsável por essa tarefa, o que também é o mais natural. O gráfico 17 mostra esses dados.

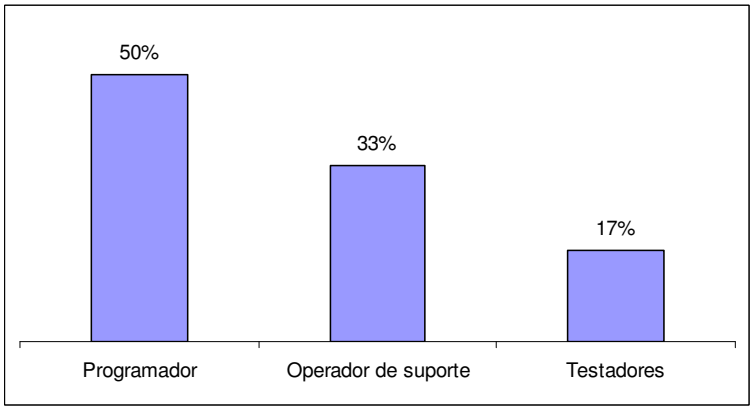

Gráfico 17: Responsáveis por implementar testes

Com os testes já implementados, é hora de executá-los. É importante que os testes sejam executados só pela equipe, para depois junto ao cliente. Caso erros sejam descobertos pela equipe, eles ainda podem ser corrigidos antes do teste com o cliente. É interessante também que a pessoa que for testar o módulo não tenha sido a mesma que o implementou, para que os erros sejam identificados com mais facilidade.

Quatorze e meio por cento das empresas afirmaram não executar testes, tanto apenas pela equipe quanto junto ao cliente. Isso gerará problemas mais tarde, quando o sistema já estiver em funcionamento. Caso o sistema pare, o cliente será prejudicado na realização de suas atividades até o erro ser corrigido.

Em trinta e seis por cento, o teste antes do sistema ser entregue é realizado pelo engenheiro de testes, o que é o mais adequado, pois essa é sua função dentro da equipe. Já os testes realizados junto ao cliente, o mesmo percentual é realizado pelo operador de suporte, já que será através dele que o cliente apontará os erros, mas a presença do engenheiro de testes seria importante, já que ele executou os testes junto à equipe. Os gráficos 18 e 19 mostram esses dados.

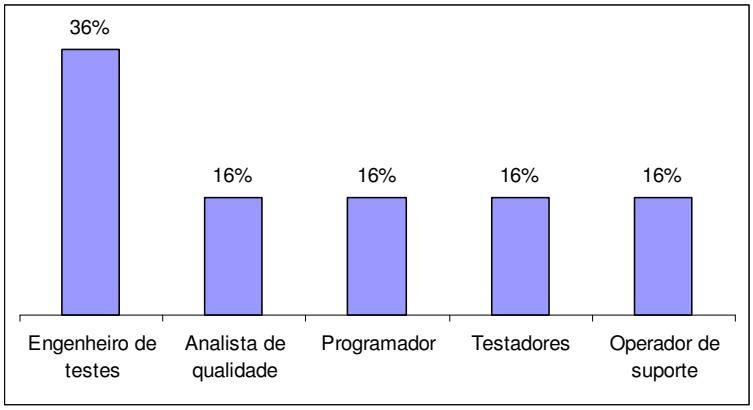

Gráfico 18: Responsáveis por executar testes

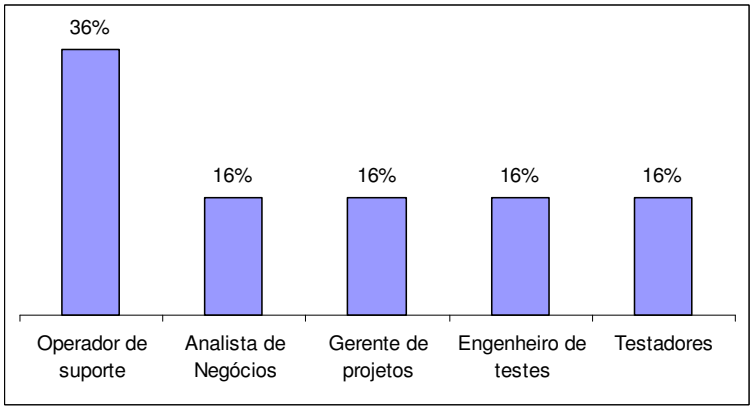

Gráfico 19: Responsáveis por executar testes com cliente

A última fase é a de manutenção do sistema, que se inicia depois da entrega para o cliente e já está em funcionamento. Nessa fase, surgem os erros que não foram descobertos na fase de testes. Todas as empresas afirmaram que a manutenção sempre acontece, seja através de treinamento dos usuários, entrega de manual explicativo ou contato, tanto por telefone quanto pela internet e também através de visitas.

Dois perfis presentes nas respostas que realizam esta atividade são o analista de negócio e operador de suporte, o primeiro com cinqüenta e sete por cento. Esses dois profissionais visitam o cliente com objetivos distintos. O primeiro irá cuidar da

negociação com o cliente, verificando se o sistema está sendo satisfatório e também se a empresa dele não precisa de mais alguma solução que a tecnologia da informação possa resolver. O segundo tratará de questões técnicas do sistema construído. Assim, o trabalho de um não interfere no do outro.

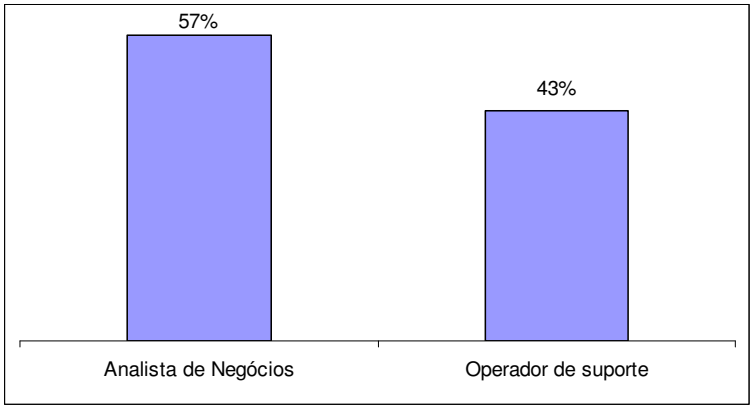

Gráfico 20: Responsáveis por visitar o cliente após entrega do sistema

4.4 USO DE FERRAMENTAS DE APOIO ÀS ATIVIDADES

Existem algumas ferramentas que têm como objetivo gerenciar projetos. Elas ajudam a ter um controle do progresso do projeto. No caso de desenvolvimento de *software*, o progresso pode ser definido como um indicador, que fornece informações sobre o andamento do projeto, referentes ao cronograma de atividades e aos serviços que devem ser fornecidos ao cliente.

É importante que em cada projeto seja estabelecido um cronograma, onde são indicadas as tarefas e os responsáveis pela sua realização, e ainda a interdependência entre elas.

O progresso pode ser avaliado verificando as atividades completas no cronograma do projeto. As diferenças entre as tarefas realizadas e as planejadas indicam o progresso do projeto. Desvios significativos em relação ao esperado indicam problemas, e medidas devem ser tomadas para contorná-los.

Cada etapa do ciclo de vida tem atividades a serem realizadas, e que podem ser quantitativamente medidas com métricas de processo. Cada atividade produz saídas tangíveis, tais como código fonte ou projeto de algum subsistema, que podem ser efetivamente monitorados com métricas de produto. Pode-se perceber claramente a evolução do projeto, monitorando essas atividades.

Na pesquisa realizada, foi possível perceber que quarenta e três por cento das empresas não utilizam nenhuma ferramenta desse tipo. Porém, das ferramentas usadas, a MSProject (Microsoft) é a mais utilizada, totalizando quarenta por cento , como mostra o gráfico 21.

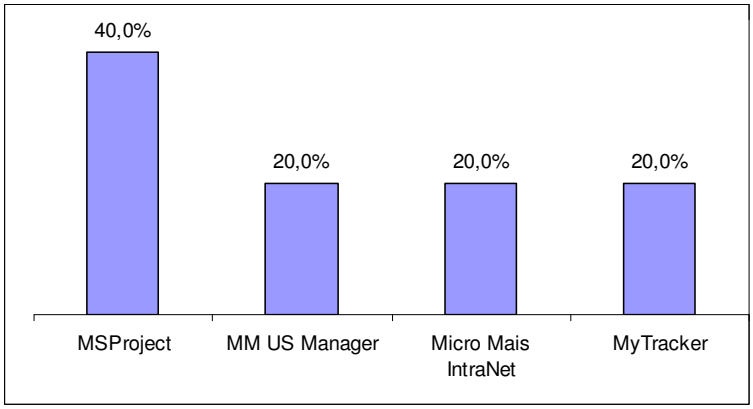

Gráfico 21: Ferramentas de Gerência de Projetos

A modelagem do sistema é de grande importância para o sucesso do projeto. São construídos diagramas, que podem ser de atividades, classes, componentes, colaboração, casos de uso, dentre outros. Assim oferece uma visão pragmática e facilitada do sistema a ser construído, pois mostra a relação existente entre as classes, indicando todos os componentes a serem construídos e os tipos de dados, dentre outras propriedades.

De acordo com a pesquisa, todas as empresas realizam essa atividade. A ferramenta mais utilizada é *Rose* (IBM), com quase quarenta e três por cento. Ela é seguida com um empate entre *Jude* (Change Vision) e *ErWin* (All Fusion), conforme é visto no gráfico 22.

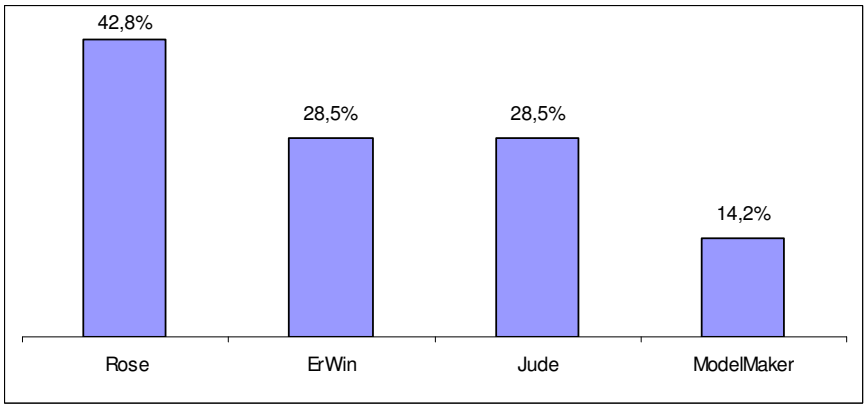

Gráfico 22: Ferramentas de Modelagem de Sistemas

Já as ferramentas de controle de versões, como o próprio nome sugere, ajuda a controlar as versões do sistema que está sendo construído, a fim de evitar problemas durante a integração dos módulos. Através do gráfico 23, com quase quarenta e três por cento, é possível notar que a ferramenta mais utilizada é a CVS (*Software* Livre), seguida da *Subversion* (Tigris), com vinte e oito e meio por cento. As empresas que afirmaram usar outras ferramentas não apontaram os nomes das mesmas.

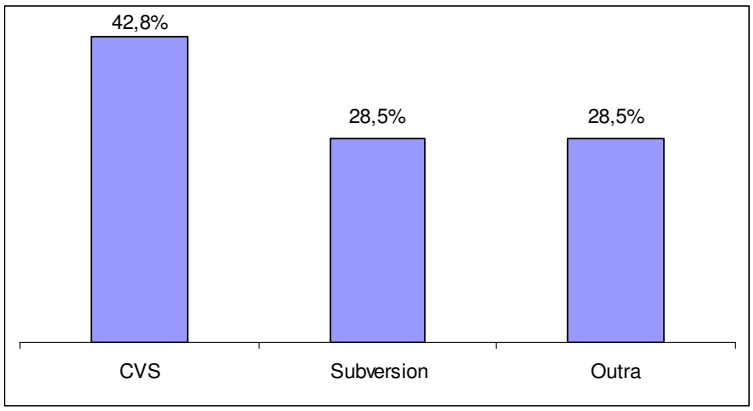

Gráfico 23: Ferramentas de Controle de Versões

O uso de uma linguagem ou outra tem muita influência no sucesso do projeto, pois cada linguagem possui suas peculiaridades, facilitando a construção para cada tipo de sistema.

Referente às ferramentas de implementação, a pesquisa mostra que um pouco mais de setenta e um por cento desenvolvem seus sistemas usando o ambiente de programação *Delphi* (Borland). Isso se explica pelo fato de a maioria das aplicações ser da plataforma *Windows* (Microsoft), e esse ambiente facilita bastante esse tipo de programação. Outras ferramentas mais usadas são as linguagens Java (Sun) e *PHP* (Open Source), com quase quarenta e três por cento cada uma. O gráfico 24 mostra esses dados e outras ferramentas usadas.

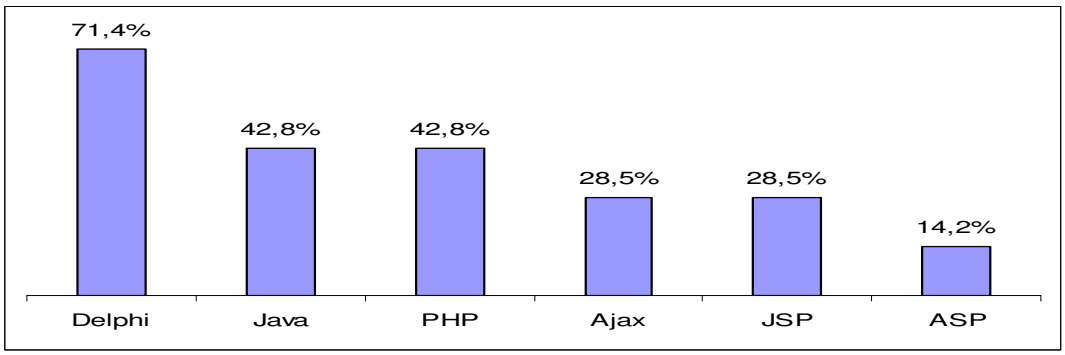

Gráfico 24: Ferramentas de implementação

Da mesma forma que a linguagem é importante, o sistema gerenciador de banco de dados também deve ser bem avaliado. As informações devem ser guardadas em um banco confiável, adequado para o tamanho dos dados. O mesmo percentual que afirmou utilizar o ambiente de programação *Delphi* (Borland) diz também utilizar o banco *Firebird* (*Interbase*). Isso se deve ao fato de essas ferramentas serem integradas. Outra ferramenta muito utilizada é a *Postgree* (*Software* Livre) (com quase quarenta e três por cento), conforme pode ser observado no gráfico 25.

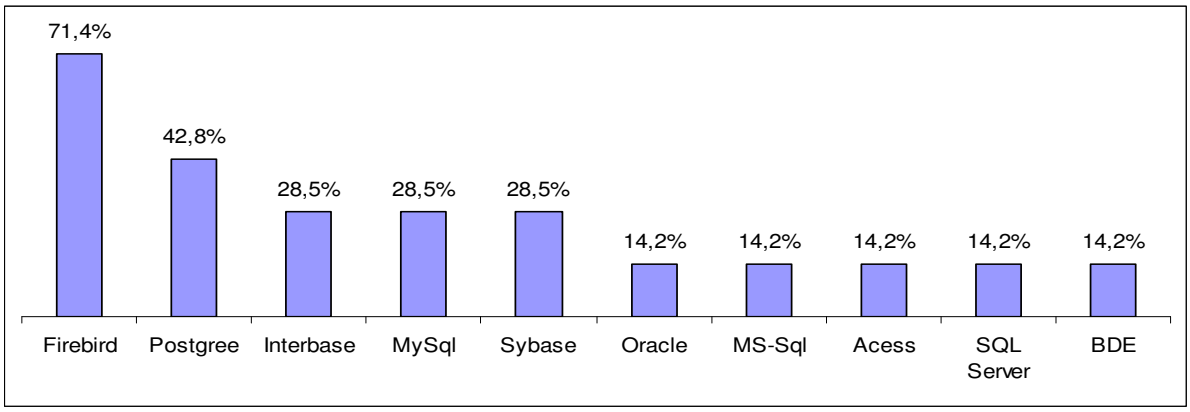

Gráfico 25: Ferramentas de Banco de Dados

4.5 CUMPRIMENTO DE PRAZOS E CUSTOS

O cumprimento do que foi combinado com o cliente é muito importante para a imagem da empresa, pois o cliente pode não querer fazer um novo contrato caso necessite de um novo sistema assim como pode não indicar a empresa produtora de sistemas, impedindo que esta adquira novos clientes e amplie sua atuação no mercado.

Os compromissos mais importantes sem dúvida são de prazo e custo. O cliente não ficará satisfeito em ter que esperar mais tempo que o combinado, pois o sucesso de seu negócio pode depender diretamente do sistema que encomendou. Ele também não vai gostar de ter que pagar a mais, pois isso significa uma nova despesa, já que seu orçamento foi calculado com base no que foi cobrado anteriormente. Ele também não teve culpa se o valor cobrado não for suficiente, a não ser que ele peça novos requisitos ao longo do projeto e que não estavam previstos quando o contrato foi feito.

O desenvolvimento de sistemas não é uma tarefa muito fácil, mesmo com a utilização de ferramentas apropriadas e adoção de metodologia, alguns problemas surgirão no decorrer do projeto. O principal deles, com setenta e um por cento, diz respeito aos prazos. Isso pode ser um reflexo tanto da não adoção de alguma metodologia de desenvolvimento quanto não utilização de alguma ferramenta de apoio ao gerenciamento de projetos. Outro problema enfrentado é a realização de testes, com cinqüenta e sete por cento. Ele vem devido à falta de realizações de algumas tarefas na fase de testes, como foi visto na seção 4.3 deste capítulo. O gráfico 26 mostra outros problemas enfrentados.

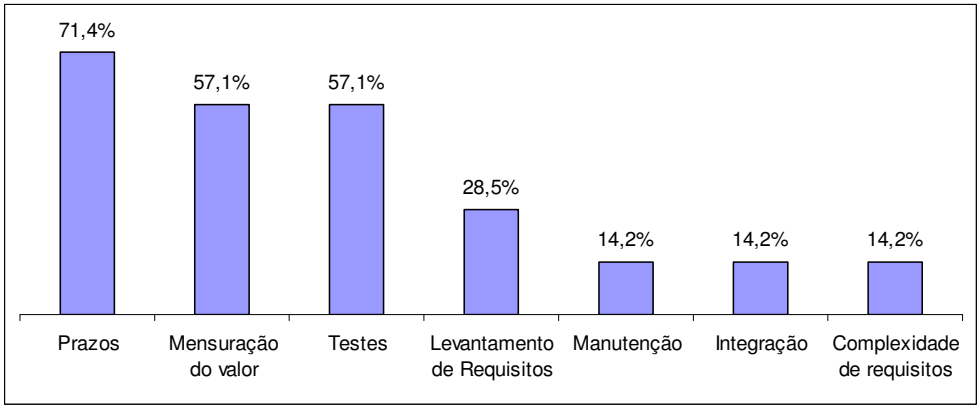

Gráfico 26: Problemas enfrentados pelas empresas

Com relação aos compromissos de prazos, o não cumprimento acaba gerando transtornos com o cliente, contribuindo assim para a sua insatisfação. Conforme visto no gráfico anterior, as empresas locais têm dificuldade em cumprir seus compromissos de prazos. Isso é uma evidência da crise do *software*, como possível conseqüência de dois fatores: tanto de algumas empresas não seguirem nenhuma metodologia adequada como também não usarem uma ferramenta que auxilie no gerenciamento de projetos, permitindo assim que as atividades sejam realizadas em sua maioria sem algum tipo de gerenciamento.

De acordo com as respostas dadas, foi possível perceber que apenas cinqüenta e sete por cento das empresas conseguem cumprir seus compromissos de prazos com facilidade, enquanto que quarenta e três por cento com dificuldade, como é mostrado no gráfico 27.

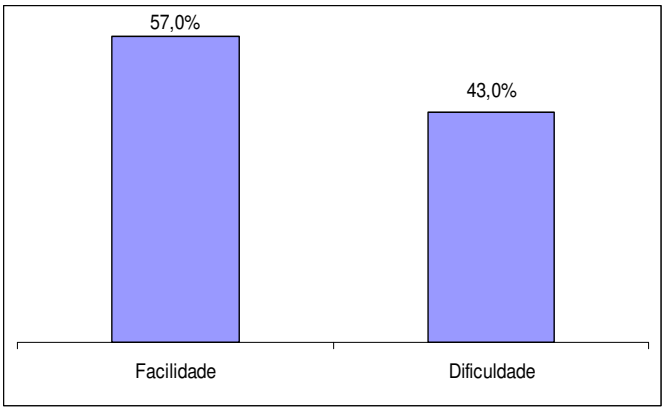

Gráfico 27: Quando prazos são cumpridos

Quando os prazos não são cumpridos, alguma medida dever ser tomada para que o problema seja contornado. As principais providências que podem ser tomadas são: reunir a equipe, aumentar a jornada de trabalho dos componentes e renegociar prazos com o cliente, com quase quarenta e três por cento em ambos os casos. Caso existisse um controle dos riscos, haveria a possibilidade de um plano de contingência que minimizasse ou reduzisse a zero os impactos. No entanto, isso não acontece de forma controlada e sistematizada. Outras medidas são mostradas no gráfico 28, como alternativas, que também não são periodicamente controladas numa lista de riscos.

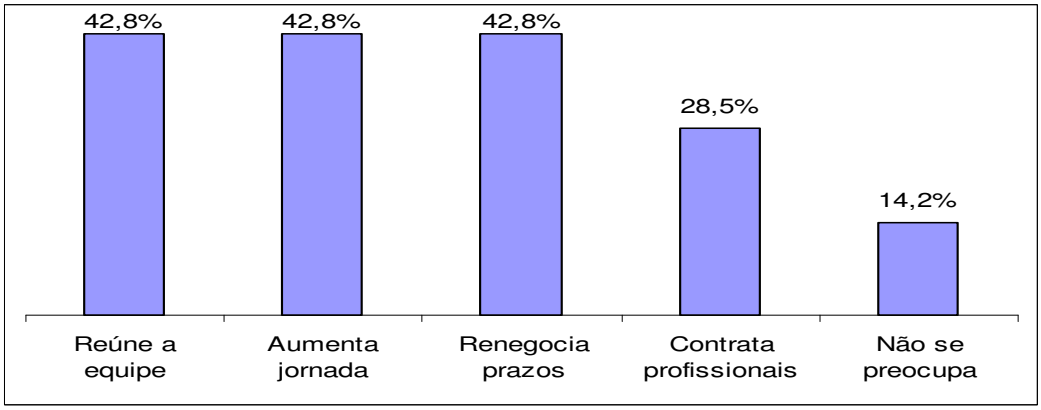

Gráfico 28: Medidas tomadas quando prazos não são cumpridos

Ao que se refere aos compromissos de custos, os resultados são melhores comparados aos de prazos. Quarenta e três por cento afirmaram que conseguem cumprir todos os compromissos e cinqüenta e sete por cento cumprem grande parte. Ainda assim, setenta e um e meio por cento dos custos cumpridos são com facilidade. Os gráficos 29 e 30 seguintes mostram esses dados.

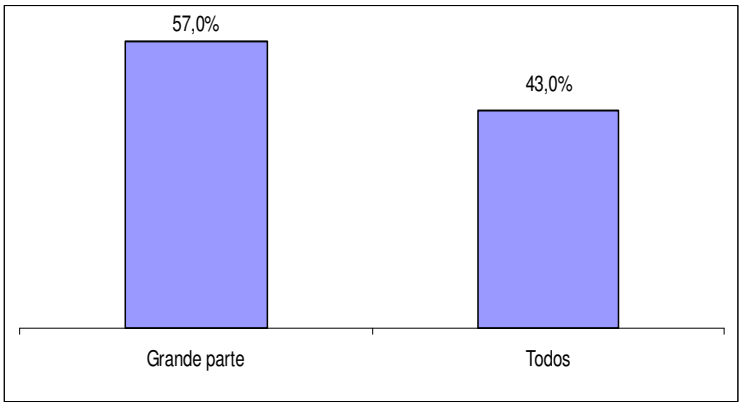

Gráfico 29: Cumprimento de custos

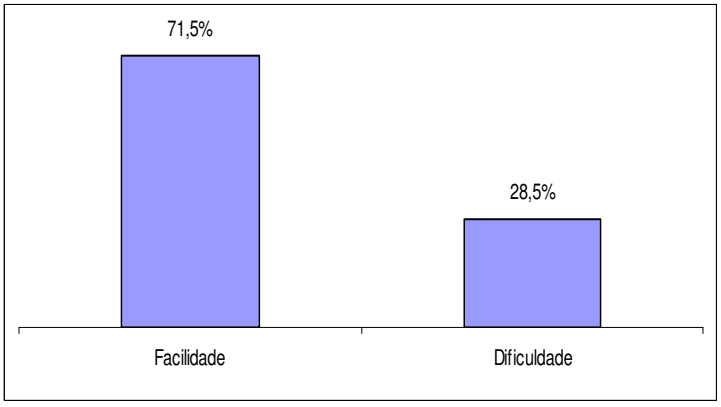

Gráfico 30: Quando custos são cumpridos

Porém, acontece de alguns compromissos de custos não serem cumpridos, sendo necessário organizar medidas como: cinqüenta e sete por cento das empresas assumem os novos custos, pois parte deles poderia ter sido evitada com um planejamento prévio; e o mesmo percentual afirma renegociar com o cliente, quando os novos custos referem-se à mudança de requisitos, o que é justo, porém apresenta um problema na especificação e levantamento de requisitos. O gráfico 31 mostra esses dados.

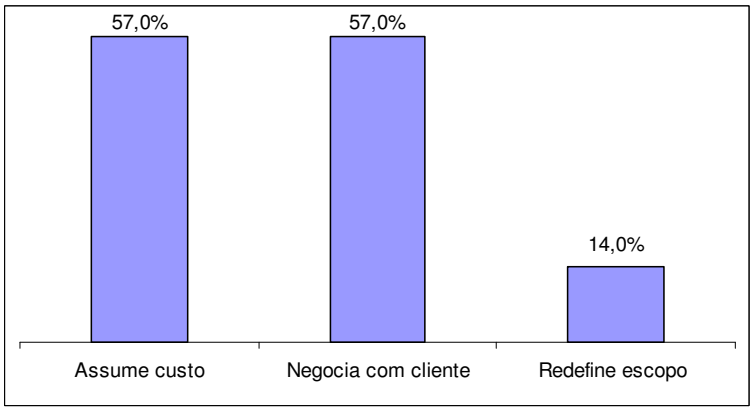

Gráfico 31: Medidas tomadas quando custos não são cumpridos

Essa facilidade apresentada no gráfico 30 em relação ao cumprimento de custos não é natural nos projetos de *software*. A pesquisa do Standish Group mostra que somente 16,2% dos projetos de software terminaram dentro do prazo e orçamentos estimados (THE STANDISH GROUP, 1995). Duas hipóteses podem explicar essa facilidade que as empresas locais encontram em cumprir seus custos. A primeira é que as empresas lucram muito com suas atividades devido aos preços altos cobrados ou possuem um número grande de clientes; ou ainda ambas as possibilidades. Uma evidência é que quando os custos não são cumpridos, mais da metade das empresas assumem o mau planejamento. Outra hipótese é a falta de concorrência. No gráfico 3 da seção 4.1, é possível perceber que existem diversos segmentos de atuação, com poucas empresas em cada um deles.

4.6 ANÁLISE DAS ATIVIDADES

O fato de não usar uma metodologia acaba comprometendo a realização das tarefas. Elas precisam ser determinadas, obedecendo a um cronograma para que o projeto não seja comprometido. Se as atividades não são repetidas, isso significa que nenhuma experiência é adquirida, e as decisões vão sendo tomadas conforme os problemas vão surgindo.

De acordo com a pesquisa realizada, apenas quase quarenta e três por cento das empresas responderam que suas atividades são determinadas. O restante afirmou que as atividades são repetidas e outras vão sendo realizadas à medida que vão aparecendo, conforme mostra o gráfico 32. Isso mostra que as tarefas são realizadas sem interdependência, não fazendo parte de um processo bem definido.

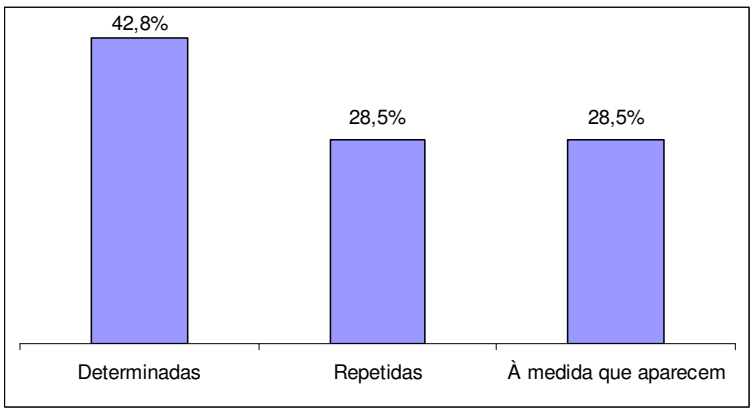

Gráfico 32: Realização de tarefas

O fato de algumas empresas realizarem suas atividades à medida que elas surgem se deve ao fato delas não gerenciarem os riscos do projeto. Quando os problemas aparecem, a equipe é surpreendida e se preocupa em resolver apenas aquele problema, sem prever os próximos que virão. A pesquisa mostra que oitenta e seis por cento das empresas não gerenciam os riscos.

Além do planejamento das atividades, é importante que elas sejam documentadas. Isso contribui bastante para a equipe saber o que realmente está sendo feito e se surgir alguma dúvida, ela pode ser facilmente resolvida consultando esses documentos. Ajuda também no caso de alguém sair da equipe durante o projeto, pois quem for assumir as funções não terá dificuldade em se integrar à equipe e realizar suas funções. As informações do projeto devem ser acessíveis para toda a equipe, assim as atividades não precisam estar diretamente ligadas às pessoas que as realizam.

O gráfico 33 mostra as tarefas que são documentadas pelas empresas, sendo que as tarefas mais documentadas são a especificação (cinqüenta e sete por cento) e codificação do sistema (quase quarenta e três por cento).

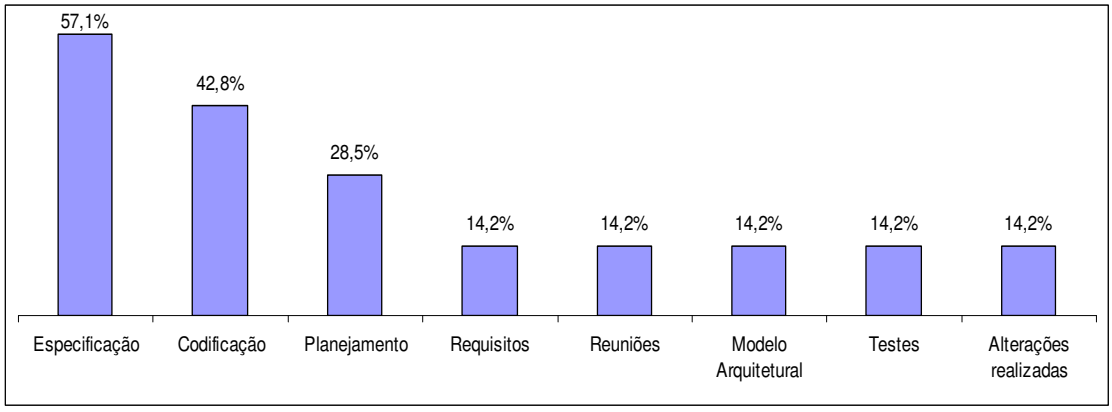

Gráfico 33: Atividades documentadas

4.7 METODOLOGIA DE DESENVOLVIMENTO

Uma questão muito importante na criação de sistemas se refere ao uso ou não de uma metodologia de desenvolvimento apropriada ao tipo de projeto e tamanho da equipe. A simples decisão de usar ou não uma metodologia pode ser um fator determinante para o sucesso do projeto, pois ela guia a execução das atividades, permitindo um planejamento, determinando os responsáveis por elas e prazos de entrega, assim como realizar reuniões para discussão do projeto.

Exemplos de metodologia são RUP (*Rational Unified Process*) e XP (*Extreme Programming*). Quando uma empresa for escolher a metodologia a ser adotada, ela deve levar em consideração o tamanho da equipe e do projeto.

De acordo com a pesquisa, cinqüenta e sete por cento das empresas afirmaadotar alguma metodologia de desenvolvimento. Porém, conforme as análises feitas neste capítulo, pode-se concluir que apenas existe a realização de tarefas que não são ordenadas entre si. Percebe-se que há falhas na prática com relação à execução dessas tarefas, logo, verdadeiramente, não há processo de desenvolvimento, pois um processo, conforme definido no capítulo 2, tem como função “produzir bens e produtos de uma forma organizada, repetitiva e mantendo a mesma qualidade” (CRUZ, 1998).

Uma dessas falhas diz respeito ao gerenciamento das atividades. Na seção anterior foi mostrado que menos da metade das empresas possuem suas atividades documentadas e determinadas. Os riscos dos projetos não são gerenciados, ou seja, nenhuma análise prévia é feita a fim de evitar problemas. Outra evidência é o fato de que os compromissos de prazos muitas vezes são comprometidos, sendo esse o principal problema apontado pelas empresas.

Das empresas que afirmaram adotar alguma metodologia, setenta e cinco por cento afirmaram usar a metodologia *Extreme Programming*, mais conhecida como XP. Conforme foi explicado no capítulo 2 dessa monografia, ela é ideal para ser aplicada em empresas de pequeno porte, como é o caso das empresas locais. Os nomes das outras metodologias não foram dados no questionário aplicado.

4.8 PADRÕES DE QUALIDADE

Conforme foi visto no capítulo 3, a competitividade entre as organizações vem crescendo cada vez mais, assim elas precisam criar vantagem competitiva. A diferenciação principal é a qualidade de processo. A busca de certificação está diretamente ligada ao avanço em relação à grande concorrência. Na época atual, melhorar os processos é de fundamental importância para que o produto final seja satisfatório, principalmente ao que se refere aos produtos de *software*.

Devido à falta de concorrência no mercado local, como foi visto na seção 4.5 deste capítulo, as empresas não se preocupam em obter certificação. Assim, os processos tendem a continuar imaturos e qualidade difícil de ser alcançada.

Dos padrões de qualidade direcionados a informática existem aqueles que avaliam o produto de *software*, levando em consideração algumas características, tais como: funcionalidade, facilidade de uso, dentre outros atributos. Um exemplo é o padrão ISO 15504, específico para sistemas de informática. Existem também os padrões que avaliam o processo de desenvolvimento de sistemas, como por exemplo, o CMMI, PSP e MPS-Br, já detalhados no capítulo 3 dessa monografia.

Desses padrões, o ISO é conhecido por todas as empresas entrevistadas. O segundo mais conhecido é o CMMI, com setenta e um por cento. Poucas empresas afirmaram conhecer o padrão MPS-Br, somente vinte e oito e meio por cento, enquanto que apenas catorze por cento delas conhecem o PSP. O que elas não sabem é que o MPS-Br e PSP são direcionados justamente às organizações de micro e pequeno porte. O gráfico 34 mostra esses dados.

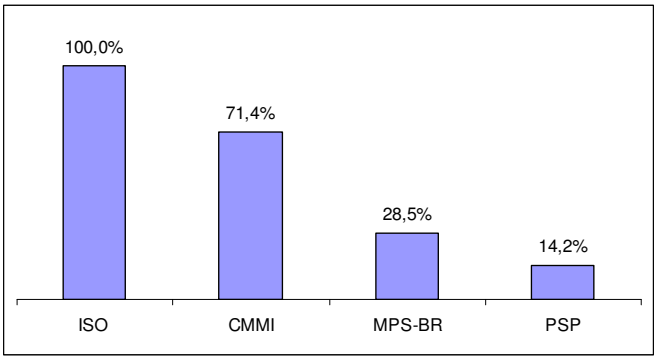

Gráfico 34: Padrões de qualidade conhecidos

Apesar do ISO ser o mais conhecido, cinqüenta e sete por cento das empresas responderam que gostariam de obter o padrão CMMI. Pela dificuldade de obtê-lo, algumas empresas acabam perdendo o interesse em obter qualquer padrão de qualidade, como mostra o gráfico 35. O ideal seria que as empresas buscassem por uma certificação, de preferência a avaliação do MPS-Br, por ser mais adequado à sua realidade.

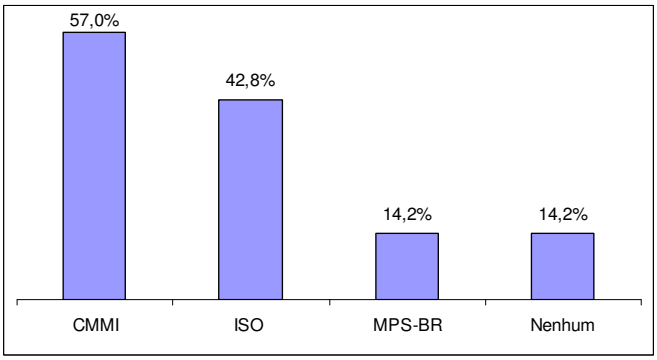

Gráfico 35: Padrões de qualidade desejados

5. CONCLUSÃO

Esse trabalho proporcionou a obtenção de mais conhecimento sobre processos de desenvolvimento e qualidade de *software*, além de ser possível conhecer a realidade das empresas produtoras de *software* de Vitória da Conquista.

O desenvolvimento de sistemas é uma tarefa árdua. Como comprovam as pesquisas do *Standish Group Report*, apenas 16.2% dos projetos de *software* são completados dentro do prazo e custo previstos (THE STANDISH GROUP, 1995)

Como uma alternativa para solucionar esse problema, surge a Engenharia de *Software*. Ela se ocupa dos aspectos da produção de sistemas desde a fase de especificação do sistema até a manutenção depois que ele entrou em operação (Sommerville, 2003).

Para que se obtenha sucesso, os processos de trabalho precisam ser bem definidos e de forma organizada. A realização das documentações é de extrema importância para que seja possível gerenciar os processos, buscando manter a mesma qualidade (CRUZ, 1998).

Existem metodologias que auxiliam o gerenciamento da produção de sistemas. A adoção de uma metodologia é fundamental para o sucesso do projeto. Cada atividade deve ser realizada corretamente e devidamente documentada para que o resultado seja satisfatório.

O uso de ferramentas específicas que auxiliam o gerenciamento de projetos facilita bastante esse trabalho. As atividades devem ser planejadas para que um cronograma seja seguido a fim de evitar atrasos. O gerenciamento dos riscos também é importante para que não haja problemas durante o desenvolvimento do projeto, evitando assim o aumento de custos e dilatação dos prazos estabelecidos com o cliente.

Existem metodologias direcionadas às empresas de pequeno porte, como exemplo, a Fábrica de *Software* III que surgiu no ambiente acadêmico da UFPE (Universidade Federal do Pernambuco). Sua principal característica é a organização e a padronização das tarefas com o objetivo de auxiliar na coordenação e formalização dos processos. Aborda as várias fases do projeto, desde a concepção até a entrega do sistema. Todas as atividades geram artefatos que servem para avaliar o progresso do projeto (AAEN; BOTTCHER; MATHIASSEN, 1997).

Chiossi e Côrtes (2001) explicaram que após a Segunda Guerra Mundial, aumentaram a capacidade de produção e diversidade dos produtos, aumentando assim a competitividade entre as organizações. Nesse ambiente competitivo, para que uma empresa se destaque no mercado, a qualidade dos seus processos é fundamental.

Assim, surgem os padrões que têm como objetivo avaliar a qualidade dos processos empresariais. Através de leitura sobre padrões que avaliam a qualidade de *softwares*, foi aprendido que existem dois tipos de padrões: os que avaliam o produto e os que avaliam o processo de desenvolvimento. Este trabalho teve como foco o segundo tipo, e foram detalhados alguns exemplos, como o CMM, PSP e MPS-Br.

O MPS-Br propõe definir um modelo de melhoria e avaliação de processo de *software* das empresas brasileiras de desenvolvimento compatível com o CMMI. Propõe também implementar o modelo a um custo acessível em todas as regiões do país, com foco nas pequenas e médias empresas (ARAÚJO et al, 2004).

Após aplicação dos questionários, foi possível perceber que algumas empresas não se preocupam o suficiente com o processo, deixando de realizar atividades importantes, como por exemplo, executar testes no sistema e gerenciar os riscos do projeto.

Outro problema é que as empresas não realizam documentações de suas atividades de forma consistente. O gerenciamento do projeto fica comprometido pela ausência de documentação de atividades importantes como planejamento, reuniões, testes e alterações realizadas. As atividades mais documentadas são a especificação e codificação do sistema.

Foi possível concluir que muitas empresas não adotam metodologia de desenvolvimento, e as que seguem alguma, não o fazem por completo. Como conseqüência, algumas atividades são determinadas ou repetidas, enquanto que as outras são realizadas à medida que aparecem, como uma operação de apagar incêndio.

Isso é uma conseqüência do fato de oitenta e seis por cento das empresas não avaliarem os riscos do projeto, com preocupação apenas em resolver os problemas que surgem, sem tentar prever os que podem aparecer. Os compromissos de prazos acabam ficando comprometidos e muitas empresas acabam cumprindo-os com dificuldade.

É importante ressaltar que os egressos do curso de Ciência da Computação são empreendedores no mercado de Vitória da Conquista de desenvolvimento de *software*. Exatamente a metade das empresas foi criada por essas pessoas e em todas elas existe pelo menos um estudante desse curso.

Mesmo assim, o conhecimento obtido sobre Engenharia de *Software* não é completamente aplicado na prática. Parte delas não adota uma metodologia de desenvolvimento, não se preocupa em obter alguma certificação, não realiza algumas tarefas da fase de testes e nem faz documentação das atividades.

Um grande diferencial entre as empresas é qualidade de seus produtos e processos. Porém, essa ainda não é a realidade do mercado de produção de *software* conquistense. Algumas empresas afirmaram não se preocupar com certificação. Isso pode ser conseqüência do fato de esse mercado ainda ser pequeno. A tendência é que essa realidade mude nos próximos anos, pois o número dessas empresas deve crescer. Muitas delas surgiram no ano de 2006, o que significa que novas ainda virão.

A despreocupação com a certificação pode ter uma explicação. A maioria das empresas afirmou conhecer o padrão CMMI, que é bastante complicado e caro para obter. Isso acaba desestimulando as empresas de pequeno porte. Porém, há preocupação em se avaliar os processos de desenvolvimento das pequenas empresas. Com esse objetivo, surgiram o MPS-Br e o PSP, só que poucas empresas locais sabem disso.

O essencial seria que as empresas começassem a se preocupar mais com a qualidade de seus processos, mesmo que inicialmente o objetivo principal não fosse uma certificação. As empresas deveriam buscar a efetiva adoção de uma metodologia, realizando por completo todas as fases do projeto, executando todas as atividades. Tarefas como gerenciamento de riscos e execução de testes deveriam ser levadas mais a sério, pois elas são fundamentais para o projeto.

Depois que os processos passassem a ser mais bem definidos e visíveis, seria interessante a busca por uma certificação do MPS-Br, começando do nível G onde os processos são parcialmente gerenciados e seguindo em diante em busca dos níveis mais altos.

Como o número de empresas vem crescendo, a tendência é que nos próximos anos esse mercado seja mais competitivo. Isso com certeza levará a preocupação com relação à qualidade, fazendo parte do cotidiano dessas empresas.

5.1. TRABALHOS FUTUROS

Como continuação desse trabalho, seria interessante propor métricas de fácil aplicação para avaliação dos procedimentos e rotinas de desenvolvimento de *software* das empresas de Vitória da Conquista. Assim, essas empresas começariam a se preocupar mais com a maturidade do processo e de qualidade, para que mais tarde fosse possível buscar uma certificação com um padrão conhecido.

Outro trabalho seria a criação de um *software* que controlasse essas métricas e apoiasse a avaliação de qualidade simplificada. Essas métricas seriam aplicadas inicialmente em uma só empresa produtora para acompanhamento e análises detalhadas das tarefas e rotinas como um estudo de caso.

6. REFERÊNCIAS

AAEN, I.; BOTTCHER, P.; MATHIASSEN, L. **Software factories.** In: Proceedings of the Twentieth Information Systems Research Seminar in Scandinavia, Oslo, 1997. Disponível em < http://www.cs.aau.dk/~larsm/Dr_Techn/Volume_II/17.pdf> Acesso em 08 set. 2006.

ARAUJO, E.; MACHADO, C.; ROCHA, A.; SALVIANO, C.; SCALET, D.; WEBER. K. **Modelo de Referência e Método de Avaliação para Melhoria de Processo de *Software*.** IV Simpósio Brasileiro de Qualidade de *Software*.

ASTELS, David. **Test-driven development:** a practical guide. Upper Saddle River, NJ: Prentice Hall PTR, 2003.

BECK, Kent. **Extreme Programming (XP) explicada:** acolha as mudanças. Rio Grande do Sul: Bookman, 2000.

BECK, Kent; FOWLER, Martin. **Planning Extreme Programming**. Boston: Addison-Wesley, 2001.

BROOKS, Frederick P. **The mythical man-month:** essays on software engineering, 20th anniversary edition. 2. ed. Reading,MA: Addison-Wesley, 1995.

CHIOSSI, Thelma C. S. ; CÔRTES, Mário L. **Modelos de Qualidade de *Software***. São Paulo: Editora da Unicamp, 2001

CRUZ, Tadeu. **Sistemas, organizações e métodos**: estudo integrado das novas tecnologias de informação. 2 ed. São Paulo: Atlas, 1998.

DEMARCO, Tom. **Slack:** getting past burnout, busywork, and the myth of total efficiency. New York: Broadway Books, 2001.

DUARTE, K.; FALBO, R.;. **Uma antologia da qualidade de *software*.** Mestrado em Informática - UFES. Vitória-ES. 2000. Disponível em < http://www.inf.ufes.br/~falbo /download/pub/Wqs2000.pdf.> Acessado em 10 jan 2007.

FÁBRICA DE *SOFTWARE* III. **Metodologia de Desenvolvimento de *Software*.** Recife: UFPE, 2003. Disponível em: <http://www.cin.ufpe.br/ ~fabrica3/homePage/ index.htm >. Acessado em 25 jan 2007.

FOWLER, Martin. **Refactoring:** improving the design of existing code. Upper Saddle River, NJ: Addison-Wesley, 2000.

GONÇALVES, José E. L. **As empresas são grandes coleções de processos.** 2000. Disponível em <http://www.fgvsp.br/rae/artigos/006-019.pdf.> Acessado em 29 jan 2007.

HARRINGTON, H. James. Business process improvement. New York: McGraw Hill, 1991.

JOSKO, J.; CÔRTES, M.; **P-CMM e outros modelos na Gestão de Pessoas.** VII Simpósio Internacional de Melhoria de Processos de *Software*. São Paulo, SP, 2005. Disponível em <http://www.simpros.com.br/upload/A04_2_artigo14181.pdf> Acessado em 20 jan 2007.

JÚNIOR, José Wilson. **Uma disciplina para a Engenharia de *Software*:** Um estudo do *Personal Software Process* (PSP). Projeto de Conclusão de Curso. Universidade Federal de Pelotas, 2000. Disponível em <http://www.ufpel.tche.br/prg/sisbi/bibct/ acervo/info/2000/Mono-JoseWilson.pdf> Acessado em 20 jan 2007.

POPPENDIECK, Mary; POPPENDIECK, Tom. **Lean software development:** an agile toolkit. Upper Saddle River, NJ: Addison-Wesley, 2003

PRESSMAN, Roger S. **Engenharia de *Software*.** São Paulo: Makron Books, 1995.

ROCHA, Roberto S. **Modelagem do Processo de Desenvolvimento e Manutenção de *Software* para a UINFOR/UESB.** 3° Seminário Estudantil de pesquisa da Escola de Administração da UFBA. Salvador, 2006.

SANTANDER, V.; VASCONCELOS, A.; **Mapeando o Processo Unificado em Relação ao CMM – Nível 2.** Recife: UFPE, 2000. Disponível em <http://www.cin.ufpe.br/~gmcs/AtHome/Mestrado/Dissertacao/Material/RUP/RUPxCMM.pdf > Acessado em 12 jan 2007.

SCOTT, Kendall. **O processo unificado explicado.** Porto Alegre, Bookman,2003.

SOFTEX. **MPS.BR - Melhoria de Processo do Software Brasileiro:** Guia geral. 2006. Disponível em: <http://www.softex.br/mpsbr/_guias/deafult.asp.> Acessado em 23 de outubro de 2006.

SOMMERVILLE, Ian. **Engenharia de *Software*.** 6 ed. – São Paulo: Addison Wesley, 2003.

THE STANDISH GROUP. The Standish Group Report - CHAOS. The Standish Group, 1995, Disponível em: <http://www.standishgroup.com/sample_research/chaos_1995_1.php >, Acessado em: 29 jan. 2007.

TELES, Vinícius Manhães. **Extreme Programming:** aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade. São Paulo: Novatec, 2004.

TELES, Vinícius Manhães. **Um Estudo de Caso da Adoção das Práticas e Valores do Extreme Programming**. Rio de Janeiro: UFRJ, IM/ DCC, 2005.

WEINBERG, Gerald M. **The psychology of computer programming**. New York: Van Nostrand Heinhold Company, 1971.

WILLIAMS, Laurie; KESSLER, Robert. **Pair programming illuminated**. Boston: Addison-Wesley, 2003.

WIKIPEDIA. **Crise do Software.** 2006. Disponível em: <http://pt.wikipedia.org/wiki/Crise_do_software>. Acessado em: 09 out. 2006.

_____. **Programação extrema.** 2006a. Disponível em: http://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_extrema. Acessado em: 09 out. 2006.

APÊNDICES

APÊNDICE A - Questionário aplicado às empresas

Universidade Estadual do Sudoeste da Bahia - UESB
Curso: Ciência da Computação
Disciplina: Projeto de Computação Supervisionado
Nome: Jamille Silva Madureira
Professor (a): Vanessa Bittencourt Xavier de Moura

QUESTIONÁRIO

Empresa: _____

Responsável pelas respostas: _____

1. Há quanto tempo a empresa atua no mercado de informática?

( ) Menos de 1 ano ( ) Entre 1 e 2 anos ( ) Entre 2 e 4 anos ( ) Mais de 4 anos

2. Qual é o ramo de atuação? (Pode marcar mais de uma opção)

( ) Comércio ( ) Indústria ( ) Serviço ( ) Outro(s)_____

2.1. Especifique o segmento: (Pode marcar mais de uma opção)

( ) Saúde ( ) Educação ( ) Confecções

( ) Calçados ( ) Automóveis ( ) Outro(s)_____

2.1.2. Existe demanda de TI nesse(s) segmento(s)?

( ) Alta ( ) Média ( ) Baixa

2.2 Há preocupação em investir na empresa?

( ) Sim ( ) Não

2.2.1. Caso a resposta seja **Sim**: *em que a empresa investe? (Pode marcar mais de uma opção)*

( ) Treinamento do pessoal da equipe ( ) Certificação

( ) Marketing da empresa ( ) Recursos de *hardware*

( ) Recursos de *software* ( ) Outro(s)_____

3. O estabelecimento da empresa é:

( ) Alugado ( ) Próprio

4. Em média, sobre hardware, responda:

Quantidade de computadores? _____

Processador: ( )Pentium IV ( )AMD Sempron ( )Atlon XP ( )Atlon 64 ( )Outro_____

Memória RAM: ( )128 MB ( )256 MB ( )512 MB ( )1 GB ( )Outro_____

HD: ( ) 40 GB ( ) 80 GB ( ) 120 GB ( ) Outro_____

Manutenção nos equipamentos: ( ) Própria ( ) Terceirizada ( ) Outra_____

5. Os computadores estão interligados por uma rede?

( ) Sim ( ) Não

5.1. Caso a resposta seja **Sim:** *a rede está conectada à Internet?*

( ) Sim (Provedor) _____ ( ) Não

6. Qual(ais) o(s) sistema(s) operacional(ais) utilizado(s) na empresa?

( ) Windows NT ( ) Windows 9X/ME ( ) Windows XP ( ) Windows 2000

( ) Unix ( ) Linux (distribuição)_____ ( ) Outro(s) _____

7. Em relação à equipe de desenvolvimento:

( ) Própria ( ) Terceirizada

7.1. Caso resposta seja **Própria:** *Ela é formada por quantas pessoas?*

( ) Apenas 1 ( ) Entre 2 e 4 ( ) Entre 5 e 8 ( ) Entre 8 e 10 ( ) Mais de 10

8. As pessoas da equipe possuem: (Pode marcar mais de uma opção)

( ) Nível Médio – Quantidade _____ ( ) Nível Superior – Quantidade _____

( ) Especialização – Quantidade _____ ( ) Mestrado – Quantidade _____

8.1. Referente aos cursos de Nível Superior, q*uais são aqueles que estão sendo ou já foram realizados pelas pessoas da equipe? (Pode marcar mais de uma opção)*

( ) Ciência da Computação ( ) Engenharia de Computação

( ) Sistemas de Informação ( ) Outro_____ ( ) Outro_____

9. Quais são as ferramentas de apoio às atividades de:

9.1. Gerência de projetos:

( ) Msproject ( ) Xplanner ( ) Outra(s):_____

9.2. Modelagem de sistemas:

( ) Rose ( ) Jude ( ) Outra(s):_____

9.3. Controle de versões de implementação:

( ) CVS ( ) Subversion ( ) Outra(s):_____

9.4. Implementação:

( ) Java ( ) C++ ( ) Delphi ( ) JSP

( ) PHP ( ) ASP ( ) AJAX ( ) Outra(s):_____

9.5. Armazenamento de Banco de Dados:

( ) Firebird ( ) Interbase ( ) Oracle ( ) MySql

( ) Postgree ( ) Outra(s):_____

10. Preencha a tabela a seguir, associando as atividades ao perfil funcional nas seguintes fases:

Perfis Funcionais	Atividades	Fases
(1) Analista de negócios (2)Analista de qualidade (3)Analista de sistemas (4) Engenheiro de testes (5) Gerente de projetos (6) Líder de equipe (7) Programador (8) Outro: (9) Outro: (10) Outro:	( ) Levantar as necessidades dos clientes	**Concepção**
	( ) Identificar requisitos	
	( ) Estimar esforço do projeto	
	( ) Elaborar plano de projeto	**Planejamento e gerenciamento**
	( ) Acompanhar e gerenciar o projeto	
	( ) Documentar as atividades	
	( ) Informar sobre a evolução do projeto	
	( ) Validar o projeto	
	( ) Definir problemas a serem resolvidos	**Desenvolvimento**
	( ) Especificar e projetar as funcionalidades	
	( ) Implementar as funcionalidades	
	( ) Elaborar plano de testes	**Testes e validação**
	( ) Implementar os testes	
	( ) Executar e avaliar os testes	
	( ) Executar testes de aceitação junto ao cliente	
	( ) Visitar o cliente	**Manutenção**
	( ) Corrigir erros encontrados	

11. Referente à fase de manutenção, ela é realizada:

( ) Periodicamente – Tempo médio_____ ( ) Apenas quando o cliente solicita

( )Outra(s)_____

12. Alguma metodologia de desenvolvimento é adotada junto ao grupo de desenvolvimento de sistemas?

( ) Sim ( ) Não

3.1 Caso resposta seja **Sim***: Qual?*

( ) XP ou instância ( ) RUP ou instância

( ) Outra(s)_____

13. Há documentação durante as atividades de desenvolvimento dos sistemas?

( ) Sim ( ) Não

13.1 Caso resposta seja **Sim***: Qual atividade é documentada?(Pode marcar mais de uma opção)*

( ) Especificação do *software* ( ) Planejamento

( ) Codificação do *software* ( ) Testes

( ) Reuniões através de atas ( ) Outra(s)_____

14. Quais os principais problemas enfrentados no desenvolvimento de sistemas? (Pode marcar mais de uma opção)

( ) Comunicação com o cliente ( ) Prazos

( ) Elaboração dos testes ( ) Integração dos subsistemas

( ) Detecção e tratamento de erros ( ) Manutenção do sistema

( ) Levantamento de requisitos ( ) Mensuração do valor da venda do *software*

( ) Outro(s):_____

15. Qual destes padrões de qualidade você conhece? (Pode marcar mais de uma opção)

( ) ISO ( ) CMMI ( ) PSP

( ) MPS-Br ( ) Outro(s):_____

16. Qual destes padrões você gostaria de obter para sua empresa? (Pode marcar mais de uma opção)
( ) ISO ( ) CMMI ( ) PSP

( ) MPS-Br ( ) Outro(s):_____

17. Existem reuniões com a equipe para discussão do andamento do projeto?

( ) Sim ( ) Não

17.1 Caso resposta seja **Sim***: Qual a periodicidade?*

( ) Diária ( ) Semanal ( ) Quinzenal ( ) Mensal

( ) Bimestral ( ) Trimestral ( ) Outra(s)_____

18. É dado algum treinamento para o cliente após a entrega do software?

( ) Sim, sempre ( ) Sim, apenas quando o cliente solicita ( ) Não

18.1 Caso resposta seja **Sim***: Através de que esse treinamento é dado?(Pode marcar mais de uma opção)*

( ) Manual explicativo ( ) Treinamento dos usuários

( ) Outro(s):_____

19. Como são as tarefas realizadas diariamente?

( ) São determinadas e cumpridas sistematicamente

( ) Algumas tarefas são repetidas periodicamente

( ) Nenhuma tarefa e/ou atividade é repetida, sendo realizadas à medida que vão aparecendo

20. Referente aos compromissos de prazos, a empresa:

( ) Consegue cumprir todos

( ) Consegue cumprir grande parte

( ) Consegue cumprir pequena parte

( ) Nunca cumpre os prazos estabelecidos

*20.1 Quando os prazos são **cumpridos**:*

( ) Na maioria das vezes são cumpridos com facilidade

( ) Na maioria das vezes são cumpridos com dificuldade

21. Quando há percepção que os prazos não serão cumpridos, o que é feito dentro da empresa em relação ao cliente? (Pode marcar mais de uma opção)

( ) Reúne a equipe para redefinição de prazos

( ) Contrata novos profissionais para a equipe

( ) Aumenta a jornada diária de trabalho dos componentes da equipe

( ) Não se preocupa em replanejar prazos, mas conversa com o cliente explicando a necessidade de mais tempo

( ) Não dá retorno ao cliente e nada se faz internamente

( ) Outro(s): _____

22. Referente aos compromissos de custos, a empresa:

( ) Consegue cumprir todos

( ) Consegue cumprir grande parte

( )Consegue cumprir pequena parte

( ) Nunca cumpre os custos estabelecidos

*22.1 Quando os custos são **cumpridos**:*

( ) Na maioria das vezes são cumpridos com facilidade

( ) Na maioria das vezes são cumpridos com dificuldade

23. Quando há percepção que os custos não serão cumpridos, o que é feito dentro da empresa em relação ao cliente? (Pode marcar mais de uma opção)

( ) Redefine escopo do projeto

( ) Diminui o tamanho da equipe responsável pelo projeto e aumenta o tempo de entrega

( ) Assume o custo do mal planejamento

( ) Negocia com o cliente e repassa o custo excedido

( ) Outro(s): _____

24. Os riscos do projeto (impacto, probabilidade de ocorrência e prioridades de tratamento) são documentados, controlados e acompanhados?

( ) Sim ( ) Não

24.1 Caso a resposta seja **Sim:** *em que fase essa documentação é feita?*

( ) No início do projeto ( ) Durante o projeto ( ) No final do projeto

25. Para o estabelecimento dos requisitos, são necessárias quantas reuniões com o cliente?

( ) 1 a 2 ( ) 3 a 4 ( ) 5 a 6

( ) Mais de 6 ( ) O contato com o cliente é constante ( ) Outro:_____

26. Nessas reuniões, qual o tipo de linguagem normalmente é usado?

( ) Linguagem técnica ( ) Metáfora ( ) Conversa informal

( ) Outra(s):_____

27. Quando a mudança de requisitos é necessária, isso normalmente ocorre porque: (pode marcar mais de uma opção):

( ) O cliente não soube explicar claramente o que realmente queria

( ) O cliente mudou os requisitos durante o projeto

( ) A equipe de desenvolvimento entendeu errado os requisitos

( ) A equipe de desenvolvimento modelou e entrevistou superficialmente

( ) Outra(s):_____

Obrigado pela sua contribuição para este projeto de pesquisa acadêmica!

APÊNDICE B - Relação das empresas entrevistadas

Creativer
Dot One Soluções em Tecnologia
Informação Consultoria e Sistemas
Micro Mais Tecnologia em Informática
Micro Process
Prodados Software Ltda.
Prosoft
WMoreira