

UNIVERSIDADE LUTERANA DO BRASIL

ULBRA – *CAMPUS* GUAÍBA

CURSO DE SISTEMAS DE INFORMAÇÃO



**IMPLEMENTAÇÃO DO PROCESSO DE TESTE
DE SOFTWARE EM EMPRESAS DE PEQUENO
PORTE
TRABALHO DE CONCLUSÃO DE CURSO I**

ALINE GOMES SANTOS

Paulo Roberto Samarani
Orientador

Guaíba, março de 2009.

DADOS DE IDENTIFICAÇÃO

Acadêmico(a): Aline Gomes Santos

E-mail: alinnegs@gmail.com

Professor(a) Orientador(a): Paulo Roberto Samarani

E-mail: paulosamarani@gmail.com

Título do Projeto: Implementação do processo de teste de *software* em empresas de pequeno porte

Período de realização: março a junho de 2009.

SUMÁRIO

1	DEFINIÇÃO DO TEMA	5
1.1	Tema.....	5
1.2	Delimitação do Tema	5
2	PROBLEMA DE PESQUISA	5
3	OBJETIVOS	6
4	JUSTIFICATIVA	6
4.1	Retorno de Investimentos em Teste.....	7
5	FUNDAMENTAÇÃO TEÓRICA	8
5.1	Definição de Empresa de Pequeno Porte	8
5.2	Tipos de Teste	9
5.3	Técnicas de Teste	13
5.4	Critérios de Teste.....	15
5.5	Modelos de Teste	17
5.5.1	<i>Modelo 3P3E</i>	17
5.5.2	<i>Teste de Software e a norma ISO/IEC 12207</i>	20
5.5.3	<i>Metodologia de teste baseada no Modelo IDEAL</i>	22
5.5.4	<i>Metodologia de teste baseada no Modelo V</i>	24
6	METODOLOGIA	27
6.1	Delineamento da Pesquisa	27
6.2	Definição da População-Alvo/Amostra ou Unidade de Análise	30
6.3	Técnicas de Coleta de Dados.....	30
6.4	Técnicas de Análise de Dados.....	31
7	RESULTADOS	33
8	TRABALHO DE CONCLUSÃO DE CURSO - II	34
9	REFERÊNCIAS	35

TABELAS E FIGURAS

Tabela 1 – Retorno do investimento em testes	7
Figura 1 – Processo de Teste 3P e 3E.....	17
Tabela 2 – Detalhamento dos processos 3P3E	18
Tabela 2 - Continuação	19
Tabela 2 - Continuação	20
Figura 2 – Fases do IDEAL	22
Figura 3 – Modelo “V”	25
Figura 4 – Processo do Trabalho	32
Tabela 3 - Objetivos específicos e procedimentos de pesquisa.....	32
Tabela 4 – Modelo 3P3E para empresas de pequeno porte.....	33
Figura 5 – Processo de teste integrado com o processo de desenvolvimento	34

1 DEFINIÇÃO DO TEMA

1.1 Tema

Implementação do processo de teste de *análise* em empresas de pequeno porte.

1.2 Delimitação do Tema

O objetivo principal do trabalho é demonstrar a possibilidade de aumentar a qualidade do *software* através da implementação de um processo de teste de *software*. Será realizada uma análise de vantagens, benefícios e custos de um *software* testado através de uma equipe especializada que utiliza uma metodologia própria de teste em empresas de desenvolvimento de *software* de pequeno porte.

2 PROBLEMA DE PESQUISA

Embora tenham surgido várias metodologias de teste, bem como os modelos de melhoria de teste, e haja resultados positivos, grande partes das empresas de pequeno porte ainda apresenta dificuldades em realizar teste de *software*. Isto ocorre por vários motivos, conforme descrito em Santana (2007). Os principais motivos são:

- Resistência a mudanças;
- Falta de recursos financeiros;
- Conflito entre prioridades entre projetos;
- Dificuldade em medir sucesso;
- Benefícios geralmente obtidos a médio e longo prazo;
- Muitas mudanças culturais e organizacionais em pouco tempo;
- Falta de uso de padrões de qualidade e falta de conhecimento nos modelos utilizados.

Dentre estes motivos, também há o fato dos modelos serem genéricos e exigirem diversas adaptações para tratar as dificuldades encontradas em um

processo de teste. Adaptar a necessidade da empresa, uma metodologia ou modelo de teste não é uma tarefa fácil, pois é necessário um conhecimento da metodologia ou modelo para realizar a escolha de quais as atividades serão realmente úteis. Esta tarefa é bastante trabalhosa e complexa devido às metodologias e modelos possuírem um escopo muito grande e um alto grau de detalhamento de cada etapa.

3 OBJETIVOS

O principal objetivo do trabalho será demonstrar, através do estudo de caso, a importância do processo de teste de *software* em um projeto de desenvolvimento de *software* e os benefícios que esse processo trará a empresa. Benefícios como: a garantia da qualidade do produto a ser entregue, obter menor custo do projeto, credibilidade com seus clientes e no mercado de TI (Tecnologia da Informação).

O objetivo do teste é garantir que o *software* seja entregue como foi especificado pelo cliente com o mínimo de defeitos. Para a realização dos testes são utilizadas algumas técnicas de verificação como inspeção, revisão de produtos, “*walkthroughs*” baseados em reuniões e “*check lists*” que auxiliam na identificação de defeitos de elaborações, descumprimento de padrões e das boas práticas.

4 JUSTIFICATIVA

Os testes são importantes para a garantia da qualidade do *software*. Identificar um defeito após a entrega do projeto gera um custo maior para a sua correção do que se o defeito fosse encontrado em fase de desenvolvimento.

Segundo Martin&McClure (1984):

Manutenção contabiliza 67% dos custos totais do *software*;

20% do orçamento é para corrigir defeitos;

25% é gasto para adaptar programas a um novo *hardware* ou *software*;

6% é gasto corrigindo documentação;

4% é gasto na resolução de problemas de performance;

42% é gasto realizando mudanças solicitadas pelos usuários.

4.1 Retorno de Investimentos em Teste

O processo de teste possui custo e exige prazos maiores em um projeto, esse é um dos principais motivos das empresas de pequeno porte não adotarem o processo de teste de *software*. Existe um mercado de muito competitivo onde as empresas oferecem projetos com baixo valor e realizados em curto tempo, o que tende a comprometer a qualidade do produto final e sua imagem no mercado.

Através da tabela 1, Émerson Rios (RIOS, 2006), demonstra que um projeto mesmo com o seu custo alto se torna mais barato e confiável após a entrega do projeto.

	PROCESSO SEM ESTRUTURA FORMAL PARA TESTES	PROCESSO COM ESTRUTURA FORMAL PARA TESTES (MANUAL)
HORAS DO PROJETO	10000	10000
ERROS ENCONTRADOS	1000	1000
INVESTIMENTO EM TESTES		
Pessoal	0,00	90.000,00
Infra Estrutura	0,00	16.000,00
TOTAL DO INVESTIMENTO	0,00	106.000,00
DESENVOLVIMENTO/TESTES		
Defeitos Encontrados	250	250
Custo de Correção	2.500,00	2.500,00
TESTES (*)		
Defeitos Encontrados	0	350
Custo de Correção	0,00	35.000,00
MANUTENÇÃO (**)		
Defeitos Encontrados	750	400
Custo de Correção	750.000,00	400.000,00
CUSTO DA QUALIDADE	752.000,00	543.000,00
RETORNO		
(*) Testes realizados pela equipe de testes		
(**) Defeitos encontrados após a implantação afetando os usuários		

Tabela 1 – Retorno do investimento em testes

Através da tabela acima podemos observar que quanto mais tempo leva-se para descobrir os defeitos, maior é o custo do que se fossem descobertos ao

decorrer do projeto. O incremento de custos poderá variar até 100 vezes do que seria necessário para corrigir o mesmo defeito mais cedo.

5 FUNDAMENTAÇÃO TEÓRICA

5.1 Definição de Empresa de Pequeno Porte

Os critérios que classificam o tamanho de uma empresa constituem um importante fator de apoio às micro e pequenas empresas, permitindo que estabelecimentos dentro dos limites instituídos possam usufruir os benefícios e incentivos previstos nas legislações.

No Estatuto da Micro e Pequena Empresa, de 1999, o critério adotado para conceituar micro e pequena empresa é a receita bruta anual, cujos valores foram atualizados pelo Decreto nº 5.028/2004, de 31 de março de 2004, são os seguintes:

Microempresa: receita bruta anual igual ou inferior a R\$ 433.755,14 (quatrocentos e trinta e três mil, setecentos e cinquenta e cinco reais e quatorze centavos);

Empresa de Pequeno Porte: receita bruta anual superior a R\$ 433.755,14 e igual ou inferior a R\$ 2.133.222,00 (dois milhões, cento e trinta e três mil, duzentos e vinte e dois reais).

Atualmente, esses critérios são adotados em diversos programas de crédito do governo federal em apoio às MPE.

É importante ressaltar que o regime simplificado de tributação - SIMPLES, que é uma lei de cunho estritamente tributário, adota um critério diferente para enquadrar micro e pequena empresa. Os limites, conforme disposto na Medida Provisória 275/05, são:

- Microempresa: receita bruta anual igual ou inferior a R\$ 240.000,00 (duzentos e quarenta mil reais);
- Empresa de Pequeno Porte: receita bruta anual superior a R\$ 240.000,00 (duzentos e quarenta mil reais) e igual ou inferior a R\$ 2.400.000,00 (dois milhões e quatrocentos mil reais).

Cada estado brasileiro possui uma variedade de conceitos critérios para classificar as micro e pequenas empresas, de acordo com a sua situação econômica e fiscal própria.

Os maiores limites de enquadramento são definidos por SP, RS, PR e BA, que adotaram R\$ 2.400.000,00 de receita bruta anual. Os municípios carecem de leis nesse sentido, sendo muito poucos aqueles que contemplam o segmento da MPE com legislações própria de fomento.

Além do critério adotado no Estatuto da Micro e Pequena Empresa, o SEBRAE utiliza ainda o conceito de número de funcionários nas empresas, principalmente nos estudos e levantamentos sobre a presença da micro e pequena empresa na economia brasileira, conforme os seguintes números:

- Microempresa:
 - I. Na indústria e construção: até 19 funcionários;
 - II. No comércio e serviços, até 09 funcionários.
- Pequena empresa:
 - I. Na indústria e construção: de 20 a 99 funcionários;
 - II. No comércio e serviços, de 10 a 49 funcionários.

Nos levantamentos que têm como fonte de dados o IBGE, as estatísticas sobre micro e pequenas empresas divulgadas pelo SEBRAE utilizam o critério acima. Nos levantamentos dos censos e pesquisas sócio-econômicas anuais e mensais o IBGE classifica as firmas segundo as faixas de pessoal ocupado total.

O conceito de "pessoas ocupadas" em uma empresa abrange não somente os empregados, mas também os proprietários. Essa é uma forma de se dispor de informações sobre o expressivo número de micro unidades empresariais que não empregam trabalhadores, mas funcionam como importante fator de geração de renda para seus proprietários (SEBRAE, 2006).

5.2 Tipos de Teste

A maioria dos processos de teste de *software* aplica teste de caixa preta, ou seja, os testadores não têm acesso ao código fonte, testando entradas e saídas de dados. Apenas os desenvolvedores possuem acesso ao código fonte em seus testes

unitários. Os testes unitários são realizados antes da liberação do *software* aos testes.

Tipos de teste de *software* que poderão ser aplicados em empresas de pequeno porte:

Teste Unitário: estágio mais baixo da escala de teste e são aplicados nos componentes de código criados, visando garantir que estes atendam as especificações, em termos de características e funcionalidades. Os testes unitários verificam o funcionamento de um pedaço do sistema ou *software* isoladamente ou que possam ser testados separadamente, podendo, inclusive, ser um programa ou um componente. Na grande maioria dos casos estes são realizados pelos desenvolvedores;

Teste de Integração: são executados em uma combinação de componentes para verificar se eles funcionam corretamente juntos, ou seja, assegurar que as interfaces funcionem corretamente e que os dados são processados em forma correta, conforme as especificações. Componentes podem ser pedaços de código, módulos, aplicações distintas, clientes e servidores. Estes testes podem ser feitos de forma incremental, onde cada módulo ou componente vai sendo incluído conseqüentemente até que todos os casos de teste possam ser testados. Este nível de testes deve ser realizado prioritariamente pelos desenvolvedores ou por uma equipe de testes.

Existem algumas estratégias para esse nível de teste:

Button-up – os testes começam agrupando os componentes, programas, módulos ou subsistemas do mais baixo nível para formar novos módulos ou subsistemas em níveis superiores.

Top-down – o inverso da estratégia anterior. Uma vantagem desta estratégia é que alguns resultados devem ser representados pelo usuário antes de completada a construção de componentes, programas, módulos ou subsistemas de níveis mais baixos.

Fluxo de Dados – a integração dos componentes, programas, módulos ou subsistemas, são feitos pelo desenho do fluxo de dados.

Funcional – a integração é feita baseada na junção de componentes, programas, módulos ou subsistemas que produzam um resultado funcional significativo para os usuários.

Bing-Bang – combinam com todos os componentes, programas, módulos ou subsistemas de uma só vez.

Teste de Sistema: são realizados pela equipe de teste, visando à execução do sistema como um todo ou um subsistema, dentro de um ambiente operacional controlado, para validar a exatidão e perfeição na execução de suas funções. Neste estágio de teste deve ser simulada a operação normal do sistema, sendo testadas todas as suas funções de forma mais próxima possível do que irá acontecer no ambiente de produção. Aqui deverão ser realizados os testes de carga, estresse, performance, usabilidade, compatibilidade, segurança e recuperação.

Teste de Aceitação: são testes finais de execução do sistema, realizados pelos usuários, visando verificar se a solução atende os objetivos de negócio e a seus requisitos, no que diz respeito à funcionalidade e usabilidade, antes da utilização em ambiente de produção.

Teste de Regressão: visa garantir que o *software* permaneça intacto depois de novos testes serem realizados. Um conjunto de dados e *scripts* deve ser mantido como “*baseline*” e executado para verificar que mudanças introduzidas posteriormente não danificarão códigos já considerados bons e aceitos. Os resultados esperados a partir do “*baseline*” deverão ser comparados aos resultados após as mudanças. As discrepâncias devem ser resolvidas antes de atingir o próximo nível de testes.

Teste de Carga: visa avaliar a resposta de um *software* sob uma pesada carga de dados, repetição de certas ações de entrada, de valores numéricos grandes, grandes e complexas “*queries*” a um banco de dados ou uma grande quantidade de usuários simultâneos para verificar o nível de escalabilidade, ou seja, o momento onde o tempo de resposta começa a degradar ou a aplicação começa a falhar. Nesses casos também poderão ser determinados os afunilamentos que possam existir em componentes específicos.

Estes testes podem ser aplicados durante os testes de sistema e são chamados de teste de estresse.

Teste Back-to-back: o mesmo teste é realizado em versões diferentes e o resultado é comparado.

Teste de Configuração: verificam se o *software* está apto a rodar em diferentes versões ou configurações de ambientes (*hardware* ou *software*), como, por exemplo, diversos *browsers* ou versões de *browsers*.

Teste de Usabilidade: verificam o nível de facilidade de uso do *software* pelos usuários. Deve ser efetuado principalmente em aplicações *Web*, onde existe muita navegação entre páginas. As telas de ajuda deverão ser avaliadas quanto ao seu conteúdo e clareza de linguagem, assim como as mensagens de erro. É um teste subjetivo, baseado na opinião dos usuários obtidas através de reuniões, entrevistas ou outras pesquisas. Não deverá ser executado por desenvolvedores ou a equipe de teste.

Teste de Instalação: Verificam o processo de instalação parcial, total ou atualização do aplicativo.

Teste de Segurança: validam a capacidade de proteção do *software* contra acesso interno ou externo não autorizado.

Teste de Recuperação: validam a capacidade e qualidade da recuperação do *software* após “*crashes*”, falhas de *hardware* ou outros problemas catastróficos.

Teste de Compatibilidade: validam a capacidade do *software* de executar em um particular ambiente de *hardware/software/sistema* operacional/rede etc.

Teste de Desempenho: visam a garantir que o sistema atende aos níveis de desempenho e tempo de resposta acordados com os usuários e definidos nos requisitos. São também chamados de teste de performance.

Teste Funcional: grupos de testes que avaliam se o que foi especificado foi implementado, normalmente servindo de base a um processo de verificação automática. Neste teste devem ser considerados os valores externos que examinam os limites do sistema.

Teste de Qualidade de Código: grupos de testes com o intuito de verificar o código fonte dos programas em consonância com padrões, melhores práticas, instruções não executadas e outros.

Teste de Alterações: visam a rastrear alterações durante o processo de teste.

Teste de Recuperação de Versões: verificam a capacidade de retornar uma versão anterior do sistema.

Teste de Interoperabilidade: avaliam as condições de integração com outros *softwares* e/ou ambientes. Por exemplo, se o sistema recebe informação de outro, se alguma ação deve ser tomada e vice-versa.

Teste de Sobrevivência (ou Confiabilidade ou Disponibilidade): avaliam a capacidade do *software* em continuar operando mesmo quando algum elemento (*software* ou *hardware*) fica inoperante, para funcionar. Por exemplo: queda de um dos servidores utilizados, queda de uma parte do Banco de Dados, etc.

Teste Estático: avaliam toda a documentação do projeto, tais como modelos, requisitos etc.

Teste Embutidos: avaliam a integração entre o *hardware* e o *software*, como por exemplo, um sinal para que um equipamento entre em funcionamento.

Teste de Verificação do Site Web: verificam problemas que possa haver no site *Web*, tais como *links* inválidos, arquivos órfãos, páginas lentas, ligações entre páginas/componentes entre outros.

Teste de Conferência de Arquivos: verificam alterações nos arquivos usados.

Teste Alfa: são executados quando o desenvolvimento está próximo a sua conclusão. Este tipo de teste é geralmente realizado por usuários ou outros usuários internos e não pelos programadores ou equipe de testes. Normalmente estes são feitos de forma descontrolada, isto é, sem nenhuma relação com o organograma de testes.

Teste Beta: são executados quando o desenvolvimento e testes estão praticamente concluídos, e o maior número de defeitos/problemas precisam ser encontrados antes da “*release*” final. Normalmente estes testes são feitos de forma descontrolada, isto é, sem nenhuma relação com os planos de teste.

5.3 Técnicas de Teste

O teste não pode mostrar a correção de um *software*, pois o domínio dos dados de entrada normalmente é infinito ou muito grande, ao ponto de ser inviável testar todas as possibilidades, o que corresponderia ao teste exaustivo, pela qual a

correção poderia ser comprovada. Outro aspecto é a limitação de tempo. Sendo assim, duas questões são fundamentais: como devem ser selecionados os dados para teste? E como decidir se um produto foi suficientemente testado? A fim de solucionar essas questões, foram estabelecidos critérios de teste, que estão incluídos nas técnicas distintas de teste.

As técnicas contemplam diferentes perspectivas do *software* e a necessidade de estabelecer uma estratégia de teste que complete as vantagens dessas técnicas se torna obrigatória.

Técnica Funcional: ou caixa preta aborda o *software* de um ponto de vista microscópico e estabelece os requisitos de teste com base na especificação, não utilizando conhecimento sobre uma determinada implementação (MYERS, 1979).

Técnica Estrutural ou caixa branca: estabelece os requisitos de teste com base em uma determinada implementação, verificando se atende aos detalhes do código solicitando a execução de partes ou componentes elementares do programa. Os critérios pertencentes a essa fase são classificados com base no fluxo de controle, no fluxo de dados e na complexidade.

Técnica com base de erros: estabelece os requisitos de teste explorando os erros típicos e comuns cometidos durante o desenvolvimento de *software*.

Teste de programa: visa encontrar erros no programa e procura dar evidência de que ele esteja de acordo com os requisitos do usuário.

Teste de especificação: tem por objetivo validar a especificação, ou seja, procurar evidências de que o *software* está de acordo com os requisitos estipulados pelo usuário.

Teste com base na especificação ou teste funcional: utiliza como base a especificação para validar o *software*. Os requisitos de teste e casos de teste são derivados da especificação e utilizados para testar um determinado programa que implementa a especificação.

As técnicas de teste são utilizadas de forma complementar, uma vez que elas detectam categorias de erros distintas. Para que a vantagem de cada uma seja bem explorada em uma estratégia de teste que leve a um processo de teste de boa qualidade, eficaz e de baixo custo.

5.4 Critérios de Teste

Um critério de teste serve para selecionar e avaliar casos de teste de forma a aumentar as possibilidades de revelar a presença de defeitos ou, quando isso não ocorre, estabelecer um nível de confiança na correção do produto. Um caso de teste é um par ordenado composto por dados de entrada e pela saída esperada. Os critérios de teste podem ser utilizados como:

Critério de adequação de casos de teste: quando a partir de um conjunto de casos de teste T qualquer, ele é utilizado para verificar se T satisfaz os requisitos de teste estabelecidos pelo critério. A análise de cobertura, assim denominada, consiste em determinar o percentual de elementos necessários por um critério de teste que foram executadas pelo conjunto de casos de teste. Com essas informações, o conjunto de casos de teste pode ser melhorado para que os elementos ainda não abordados sejam testados com a adição de novos casos de teste.

Critério de geração de casos de teste: quando o critério é utilizado para gerar um conjunto de casos de teste T adequado por construção ao critério.

Pode-se dizer que as propriedades mínimas que um critério de teste deve possuir são:

- Garantir, do ponto de vista do fluxo de controle, a execução de todas as transferências de fluxo de controle;
- Exigir do ponto de vista do fluxo de dados, ao menos um uso de todo resultado computacional;
- Exigir um conjunto de casos de teste finito;

Cada técnica de teste mencionada anteriormente possui diversos critérios de teste, segue:

✓ Técnica Funcional

Particionamento de equivalência: divide o domínio de entrada de um programa em classes de equivalência, a partir das quais os casos de teste são derivados. Tem por objetivo minimizar o número de casos de teste, selecionando apenas um caso de teste de cada classe, pois em princípio todos os elementos de uma classe devem se comportar de maneira equivalente. São definidas classes que se aproximam

dessa característica, pois segundo Myers (1979) é impossível caracterizar exatamente as classes de equivalência.

Análise do valor limite: complementa o critério anterior, exigindo cada caso de teste nos limites de cada classe de equivalência. Segundo Pessman (2000), os erros costumam ocorrer com maior frequência nos limites dos domínios de entrada, o que torna esse critério relevante para o teste funcional.

Grafo de causa-efeito: Verifica o efeito combinado de dados de entrada. As causas (condições de entrada) e os efeitos (ações) são idênticas e combinados em um grafo a partir do qual é montada uma tabela de decisão e, a partir desta, são derivados os casos de teste e saídas.

✓ Técnica Estrutural

Critérios com base na complexidade: Utiliza informações sobre a complexidade do programa para determinar os requisitos de teste. O critério mais conhecido desta classe é o Critério de McCabe, que utiliza a complexibilidade ciclomática para derivar os requisitos de teste.

Critérios com base em fluxo de controle: usam apenas características do controle de execução do programa, como comandos ou desvios, para determinar quais estruturas são necessárias. Os mais conhecidos são os critérios *Todos-Nós*, *Todos-Arcos* e *Todos-Caminhos*. O critério *Todos-Nós* exige que durante a execução do programa seja executado, pelo menos uma vez, cada vértice do grafo de fluxos de controle. O Critério *Todos-Arcos* exige que cada aresta do grafo seja executada pelo menos uma vez. O critério *Todos-Caminhos* é, em geral, impraticável e requer que todos os caminhos possíveis do programa sejam executados ao menos uma vez.

Critérios com base no fluxo de dados: associa ao grafo do fluxo de controle informações sobre o fluxo de dados de programa, isto é, explora associações entre pontos do programa em que é atribuído um valor a uma variável (chamado de definição de variável) e pontos em que esse valor é utilizado (chamado referência ou uso da variável). Com base nessa associação são determinados os caminhos a serem testados.

✓ Técnica com Base em Erros

Semeadura de erros: alguns erros são inseridos artificialmente no programa e durante o teste. Dentre os erros encontrados, verificam quais os erros naturais e os artificiais. Calculando a proporção de erros artificiais pelos naturais obtêm-se uma indicação de números naturais de erros ainda existentes no programa .

Análise de mutantes: tem por objetivo avaliar a quantidade de um conjunto de casos de teste T em relação a um programa P. Para isso, utiliza um conjunto de programas ligeiramente diferentes de P, denominados mutantes de P, e busca obter um conjunto de casos de teste que consiga revelar as diferenças de comportamento existentes entre P e seus mutantes.

5.5 Modelos de Teste

No processo de teste são indispensáveis: uma metodologia aderente ao processo de desenvolvimento, pessoal qualificado para a execução dos testes, um ambiente adequado em que consiga simular a realidade do cliente e ferramentas adequadas que irão auxiliar os profissionais de forma segura e rápida.

5.5.1 Modelo 3P3E

Na figura abaixo a metodologia 3P3E:

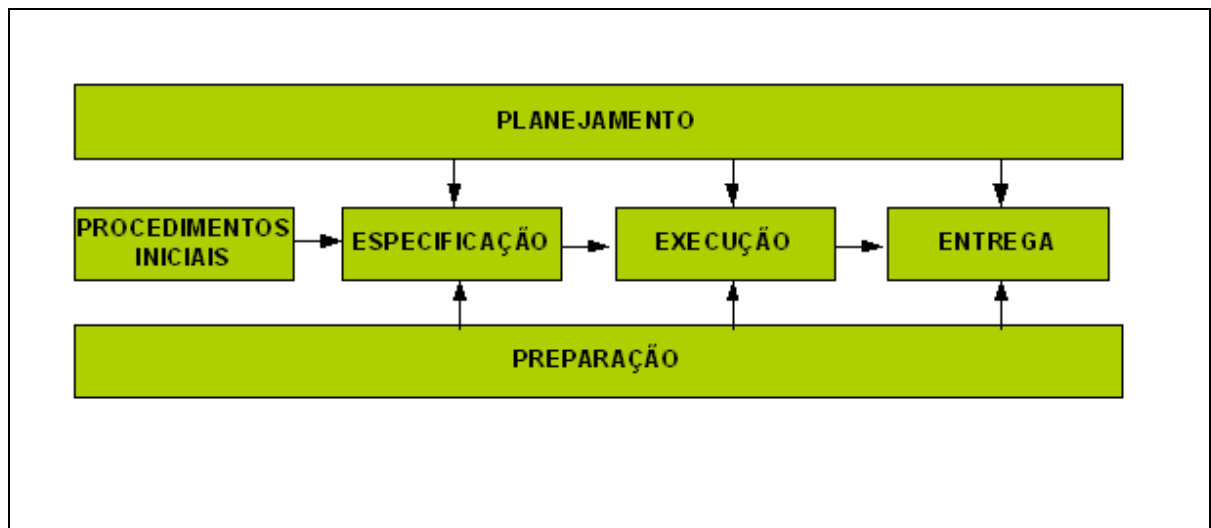


Figura 1 – Processo de Teste 3P3E

A tabela abaixo demonstra cada etapa do processo detalhadamente:

ETAPA	SUBETAPA	INSUMO	PRODUTO
Procedimentos Iniciais	Elaborar o Guia Operacional de Teste	Requisitos do Negócio; Modelos de Dados; Diagramas de Fluxo de Dados; Diagramas de Contexto; Outros documentos de desenvolvimento.	Guia Operacional de Testes
Planejamento	Estabelecer uma Estratégia de Testes	Requisitos do Negócio; Modelos de Dados; Diagramas de Fluxo de Dados; Diagramas de Contexto; Documento de planejamento do sistema; Guia Operacional de Testes (GOT).	Estratégia de Teste; Análise de Risco do Projeto de Teste.
	Estabelecer Plano de Teste	Estratégia de Testes; Análise de Riscos do Projeto de Testes; Necessidades de Dados de Teste; Planejamento do Sistema Desenvolvido;	Plano de Testes; Análise de Risco do Projeto de Teste.
	Revisar Estratégia de Testes	Requisitos de Negócio do Sistema; Estratégia de Testes.	Estratégia de Teste Revisada.
	Revisar Plano de Testes	Estratégia de Testes; Plano de Testes.	Plano de Teste Revisado.

Tabela 2 – Detalhamento dos processos 3P3E

ETAPA	SUBETAPA	INSUMO	PRODUTO
Preparação	Adequar o projeto de testes à Gerência de Configuração e/ou de controle de mudanças.	Arquitetura do ambiente de desenvolvimento; Arquitetura do ambiente de produção; Ferramentas e procedimentos de Gerência de Configuração e de mudança.	Registro e controle das diversas versões do produto: funcional, desenvolvimento, produto e operacional.
	Disponibilizar infra-estrutura e ferramentas de teste.	Estratégia de Testes; Arquitetura básica do ambiente de desenvolvimento; Arquitetura básica do ambiente de produção; Ferramentas e procedimentos de Gerência	Infra-estrutura e ferramentas de testes disponíveis para a equipe de testes.
	Disponibilizar Pessoal	Estratégia de Testes; Plano de Testes; Ferramentas de Teste; Definição de Ambiente de Teste;	Equipe de testes definida e capacitada.
	Elaborar Casos de Teste	Estratégia de Testes; Plano de Testes; Documentação Técnica do Sistema; Necessidades de Dados de Teste; Posição quanto aos testes já realizados.	Casos de Teste; "Scripts" de Testes (se usar ferramenta); Especificação das necessidades de dados de teste.
	Elaborar Roteiros de Teste	Casos de Teste; Planos de Teste; Fluxo de execução dos programas previstos pela equipe de desenvolvimento.	Roteiros de Teste.

Tabela 2 - Continuação

ETAPA	SUBETAPA	INSUMO	PRODUTO
Execução	Preparar Dados de Teste	Casos de Teste " <i>scripts</i> " de teste; Roteiros de Teste; Documentação do Sistema; Especificação das necessidades de dados de teste; Processos de criação de	Bases/Arquivos de Teste Disponíveis.
	Executar Testes	Roteiro de Testes; Casos de Testes; " <i>Scripts</i> " de teste; Resultados Esperados.	Resultados dos Testes; Relatórios de Defeitos Encontrados; Ajuste do Material de Testes.
	Solucionar Ocorrências de Testes	Relatórios de Defeitos com o status a resolver; Resultados dos Testes.	Relatórios de Defeitos encontrados com status resolvido ou a avaliar.
	Acompanhar a execução dos Casos de Testes	Relatórios de Defeitos (resumo); Resultados dos Testes (resumo); Estratégia de Testes; Plano de Testes; Casos de Testes; Roteiros de Testes.	Análise do andamento dos Casos de Teste.
Entrega	Elaborar Relatório Final	Análise dos Resultados dos Testes; Estratégia dos Testes; Resultados dos Testes; Relatórios dos Defeitos (resumo); Plano de Testes.	Relatório Final dos Testes.

Tabela 2 - Continuação

5.5.2 Teste de Software e a norma ISO/IEC 12207

A atividade de teste de *software* esta inserida nos processos fundamentais do ciclo de vida da norma ISO/IEC 12207 (ISO, 1995), mais especificadamente no processo de desenvolvimento.

A primeira atividade relacionada ao teste de *software* é a de codificação. O teste de *software* no momento do desenvolvimento tem como objetivo testar cada unidade do *software* e da base de dados, assegurando que satisfaçam os requisitos estabelecidos. A fase de teste que deve ser conduzida é o teste de unidade no qual

as técnicas mais utilizadas a estrutural e a com base em erros, muito embora o uso de critérios da técnica funcional possa enriquecer essa atividade, dando o aspecto complementar entre elas.

As próximas atividades no processo de desenvolvimento que envolve a atividade de teste são integração do *software* e teste de qualificação do *software*. Portanto a fase que deve ser conduzida é o teste de integração, no qual, além da técnica funcional, poderiam ser usados outros critérios estruturais com base na estrutura do *software* ou mesmo critérios mais rigorosos (DELAMARO, 1997).

As outras atividades do processo de desenvolvimento relacionadas à atividade de teste são integração do sistema e teste de qualificação do sistema. Nesse momento a fase de teste está em andamento é o teste do sistema, no qual a técnica mais utilizada é a funcional, uma vez que o objetivo é mostrar que o *software* atende aos requisitos estabelecidos.

Além do processo de desenvolvimento, a atividade de teste também está presente no processo de operação da norma 12207, que trata de novas versões de *software* depois que este já está em operação. Nesse caso as fases de teste devem se repetir para que a nova versão seja adequadamente testada, além de testes de regressão que devem ser aplicados para garantir que as partes do *software* que permaneceram inalteradas continuem funcionando bem com as mudanças ocorridas (ROCHA, 2001).

Todos os procedimentos e resultados de este devem ser documentados. O produto em teste e os resultados da atividade de teste devem ser avaliados conforme diretrizes preestabelecidas. Exemplo: Após o teste de unidade, os seguintes aspectos podem ser considerados:

- Rastreabilidade para os requisitos e projeto;
- Consistência externa com os requisitos e com o projeto;
- Consistência interna entre os requisitos do produto;
- Abrangência do teste;
- Adequação dos métodos e padrões de codificação utilizados;
- Viabilidade de integração e teste do produto;

- Viabilidade de operação e manutenção.

5.5.3 Metodologia de teste baseada no Modelo IDEAL

O modelo IDEAL tem como objetivo fornecer um guia para direcionar iniciativas de programas de melhoria através de um longo e integrado plano para iniciação e gerenciamento. Este modelo é definido por 5 fases: Iniciação, Diagnóstico, Estabelecimento, Ação e Lições Aprendidas.

As principais atividades de cada fase são exibidas na figura 2.

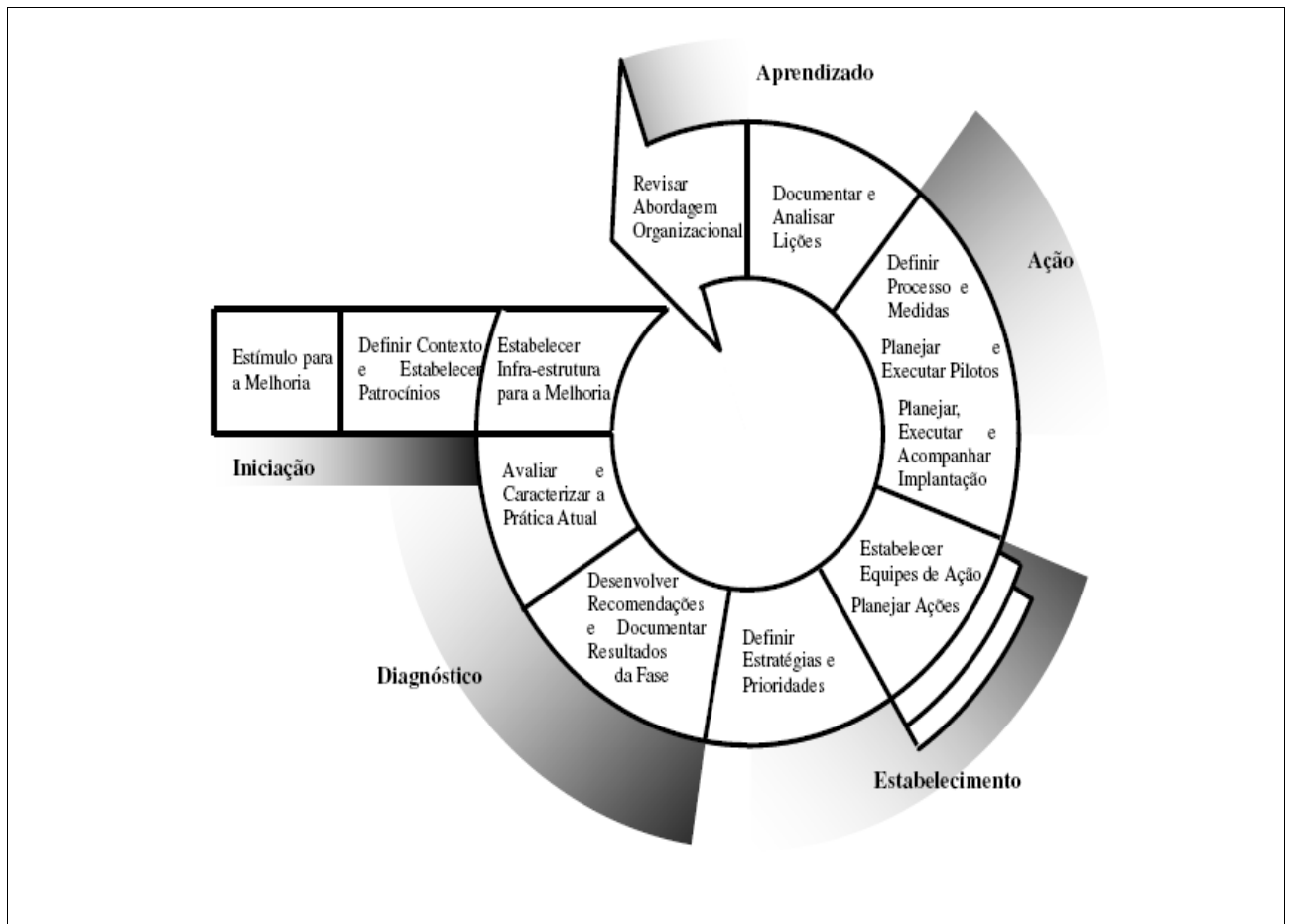


Figura 2 – Fases do IDEAL

✓ Iniciação

O objetivo dessa etapa é criar um plano de melhoria do processo de teste. Nessa etapa será definida a infra-estrutura, responsabilidades e recursos (humanos e físicos).

Serão definidas as metas a serem alcançadas através do processo de teste baseadas nas necessidades de negócios da empresa. As metas serão refinadas

para posterior utilização na criação de planos e casos de testes na fase de Estabelecimento.

✓ Diagnostico

Objetivo: Estabelecer um *baseline* do estado atual da empresa. Serão utilizados os resultados, recomendações dos validadores e outras iniciativas de melhorias para o plano de ação.

✓ Estabelecimento

As decisões de implementação da empresa é priorizada e são desenvolvidas soluções para as estratégias.

A versão inicial do plano de melhoria do processo de teste será elaborado de acordo com a visão da empresa, plano estratégico de negócios, lições apreendidas de experiências passadas.

Durante esta fase, objetivos mensuráveis definidos a partir de objetivos gerais que foram definidos na fase de iniciação, estes objetivos mensuráveis serão incluídos na versão final do plano de ação do programa de melhoria do processo de teste. Métricas necessárias para monitorar o processo são definidas, os recursos são obtidos e treinamentos fornecidos para o grupo de trabalhos técnico. O Plano de ação definido irá guiar as atividades de melhoria endereçando a prioridade das recomendações da fase de diagnostico. Também durante essa fase *templates* de plano de ação são criados e disponibilizados para os grupos de trabalhos técnicos.

✓ Ação

Nessa fase soluções endereçadas para os pontos de melhoria diagnosticados durante a fase de diagnostico são criadas, testadas e implantadas na empresa. Planos serão desenvolvidos para execução de pilotos a fim de testar e avaliar os processos novos alterados.

Depois que os pilotos são realizados com sucesso e determinada aceitação de adoção quanto ao novo/alterado processo, a implantação, institucionalização e planos para extensão são desenvolvidos e executados.

✓ Aprendizagem

O objetivo dessa fase é fazer com que o próximo passo atrás do modelo IDEAL seja mais efetivo. Neste momento, soluções já foram desenvolvidas, lições já foram aprendidas, métricas coletadas e objetivos alcançados. Os artefatos são adicionados à base de dados do processo que irá se tornar uma fonte de informação para as pessoas envolvidas no próximo passo do modelo.

5.5.4 Metodologia de teste baseada no Modelo V

O conceito do ciclo de vida dos testes é ilustrado na figura 2. A figura mostra que os processos de desenvolvimento e teste têm início simultaneamente: a equipe que desenvolve o sistema inicia o processo de desenvolvimento do sistema, e a equipe que está conduzindo os testes começa o planejamento do processo de teste. Ambas as equipes começam no mesmo ponto, usando as mesmas informações. A equipe de desenvolvimento é encarregada de capturar e documentar os requisitos com o propósito de desenvolver o sistema, e a equipe de testes usa os mesmos requisitos com o propósito de testar o sistema. Nos pontos apropriados do processo de desenvolvimento, a equipe de teste utilizará técnicas de teste como base para avaliar os produtos do processo de desenvolvimento, a fim de descobrir defeitos.

Segundo Molinari (2008), no modelo “V” de testes, os procedimentos de FAZER e CONFERIR converge do início até o fim do projeto. O grupo que FAZ trabalha com o objetivo de implementar o sistema, e a equipe que CONFERE, simultaneamente executa procedimentos de teste visando minimizar ou eliminar riscos. Com esse enfoque, se os grupos trabalharem juntos e de maneira integrada, o alto nível de riscos que caracteriza os projetos de desenvolvimento de *software* irá descrever a um patamar aceitável que permitirá a conclusão bem sucedida do projeto.

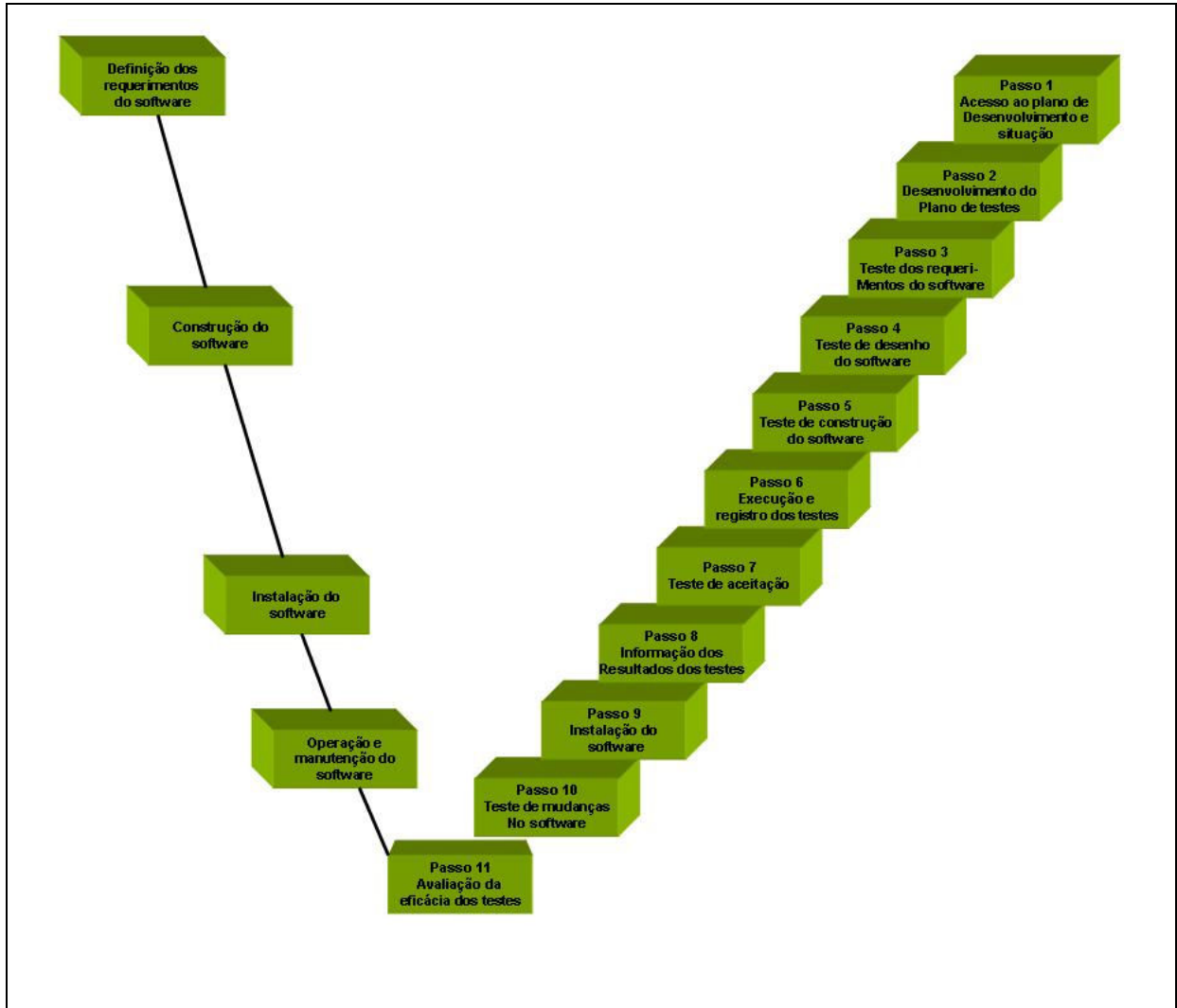


Figura 3 – Modelo “V”

A parte esquerda do modelo “V” representa o ciclo de desenvolvimento de *software*, e o lado direito ilustra alguns dos passos do processo de teste de *software*.

Os primeiros cinco passos usam a técnica de verificação como o principal meio para avaliar a correção dos produtos do desenvolvimento de *software*. Por outro lado, a técnica de validação serve para testar o *software* durante as atividades que vão desde a construção até a implantação. Os resultados de ambos devem ser registrados na documentação dos testes. A validação e a verificação devem ser usadas para o desenvolvimento e a manutenção dos *softwares*.

Alguns passos desse processo de teste:

1. Acesso ao plano de desenvolvimento: pré-requisito para a construção do plano de teste. Durante este passo, os testadores verificarão se o plano de

desenvolvimento está completo e correto. Com base nesse plano, será possível estimar a quantidade de recursos necessários para testar a solução a ser implementada.

2. Desenvolvimento do plano de teste: a preparação do plano de teste segue os mesmos padrões de preparação do plano de desenvolvimento a estrutura dos planos é a mesma, mas o conteúdo variará em função do grau de risco associado com o *software* que esta sendo desenvolvido.

3. Inspeção ou teste de requisitos de *software*: avaliação dos requisitos do *software* mediante o uso da técnica de verificação. Requisitos incompletos, indexados ou inconsistentes levam ao insucesso de boa parte do desenvolvimento de *software*.

4. Inspeção ou teste de desenho do *software*: avaliação do desenho (interno ou externo) do *software* através da técnica de verificação. O interesse da equipe de teste estará concentrado em verificar se o desenho atinge os objetivos dos requisitos, bem como se é eficaz e eficiente para operar no *hardware* previsto.

5. Inspeção ou teste de construção do *software*: o método escolhido para construir o *software* a partir do desenho do sistema determinará o tipo e a extensão dos testes que serão necessários. Quanto mais a construção se torna automatizada, menos testes serão requeridos durante esta fase.

6. Execução dos testes: envolve testar o código em estado dinâmico. A abordagem, as ferramentas e os métodos especificados no plano de teste serão empregados para validar o atendimento dos códigos executáveis aos requisitos do *software* e suas especificações de desenho.

7. Teste de aceitação: avaliação da aplicabilidade e usabilidade do *software* pelos usuários. Além dos requisitos documentados, os usuários costumam a testar outras funções não documentadas e suas expectativas. Essas situações precisam ser avaliadas com cuidado para identificar se devem ser considerados “erros” ou “mudanças”. De modo geral, os testes de aceitação devem ser orientados para avaliar se o *software* está apto a ser implantado com o nível de erros ainda não corrigidos.

8. Informação dos resultados de teste: a informação sobre os testes é um processo contínuo. Pode ser verbal (não recomendado) ou escrito (formalizado),

porém é importante que os defeitos e os tópicos envolvidos sejam relatados aos setores envolvidos o mais rápido possível, de modo que as correções sejam feitas com o menor custo.

9. Teste de implantação de *software*: visa verificar a interoperabilidade com o sistema operacional, com outros *softwares* relacionados e com os procedimentos operacionais. O resultado vai determinar se o *software* está ou não em condições de ser implantado no ambiente de produção.

10. Teste de mudanças no *software*: embora sejam consideradas o décimo passo as atividades dessa fase cobrem as mudanças durante o processo de implementação e aquelas que irão ocorrer após a implementação do *software*.

11. Avaliação da eficácia dos testes: As melhorias no processo de teste podem ser verificadas com maior exatidão pela avaliação da eficácia dos testes ao término de um projeto. Deve ser realizada pelos testadores, porém envolve os desenvolvedores, usuários e outros profissionais inseridos no processo de qualidade.

6 METODOLOGIA

Este capítulo apresenta o método de pesquisa adotado para atingir os objetivos propostos no trabalho.

6.1 Delineamento da Pesquisa

A investigação que se pretende realizar consiste em uma pesquisa qualitativa observacional. Segundo Triviños (2001), a pesquisa qualitativa é a melhor forma de compreender preceitos que fundamentam a subjetividade humana; as interpretações a respeito do tema proposto são particulares, cabendo ao investigador reunir indícios recorrentes e não generalizações. Em consonância, Stake (1998) entende que a atribuição real do estudo de caso é a especificação e não a generalização. Utiliza-se um caso particular e compromete-se a conhecê-lo bem. Os estudos de casos são a estratégia preferida ao se questionar “como” ou “por que” sobre determinado assunto de pesquisa, quando, por exemplo, o foco é um fenômeno contemporâneo dentro de algum contexto da vida real (YIN, 1994).

Alguns fatores como: requisitos do sistema, prazo de entrega, escopo, riscos e custos do projeto estão diretamente ligados à qualidade de software, pois quanto mais tempo para realizar os testes maiores os custos, quanto menor o prazo maior o risco e por isso deve-se ter uma estratégia de testes bem definida para o sucesso do projeto e uma maior qualidade do *software* (RIOS, 2006). Diante deste cenário, fez-se necessária a elaboração de uma abordagem simples e ágil que deverá atender às necessidades das empresas de pequeno porte, melhorando o seu processo de desenvolvimento de *software*, adotando o processo de teste de *software* com todas as limitações apresentadas acima para conseguir competir no mercado do qual elas fazem parte.

A abordagem proposta irá guiar as empresas de maneira simples e ágil a implantar o processo de teste de *software*, possibilitando assim uma melhor condição para estas alcançarem níveis melhores de qualidade em seus produtos. O processo de teste de *software* será guiado pelo modelo 3P3E no qual Emerson Rios descreve detalhadamente o processo em duas de suas bibliografias (RIOS, 2006 e RIOS, 2007). A empresa XXX, onde a autora exerce a atividade de analista de teste utiliza o modelo 3P3E com algumas adaptações para atender algumas peculiaridades da empresa. Os projetos em que o modelo foi utilizado caracterizava-se por possuir prazos suficientes e alto grau de risco onde a ocorrência de um defeito acarretaria em uma grande perda financeira. Através deste método a empresa XXX consegue corresponder à expectativa de seus clientes mantendo a qualidade de seus produtos antes e após a entrega.

A metodologia 3P3E é baseada em riscos, ou seja, são avaliados os riscos e executados os testes com o intuito de eliminar os erros e minimizar os riscos. A base de todo o processo de teste será a documentação (Estratégia de Testes e Plano de Testes). Emerson Rios ressalta um ponto importante em utilizar a metodologia 3P3E, a metodologia possibilita que o processo de teste seja enquadrado nas áreas de conhecimento do PMBOK (gerência de escopo, gerência de custos, gerência de prazo, gerência de contratação, e etc.).

Através da estimativa do tamanho do projeto (de testes) poderá ser estimado o esforço que será despendido neste trabalho conduzindo um planejamento de testes adequado, envolvendo prazos, custos e riscos. Utilizando os indicadores históricos

poderão ser definidas, gradativamente, metas de qualidade e de redução de custo cada vez maiores.

O ciclo de vida de testes pressupõe que sejam realizados testes ao longo de todo o processo de desenvolvimento. Em determinados pontos os produtos intermediários do ciclo de desenvolvimento deverão ser revisados com o objetivo de criar as condições necessárias para uma correta implementação, procurando-se identificar defeitos o mais cedo possível.

Os ciclos de vida de testes e desenvolvimento são totalmente interdependentes, mas o ciclo de testes é dependente da conclusão dos produtos das atividades do ciclo de desenvolvimento. A aplicação desses ciclos de vida e de suas interdependências, na ordem de execução de suas atividades, com a conseqüente geração dos produtos previstos, é um pré-requisito para viabilizar o processo de desenvolvimento do *software*.

As atividades do ciclo de vida só podem ser bem realizadas se forem bem executadas por um grupo formalmente definido para fazer testes. Esse grupo pode ser:

- Interno: formado por profissionais pertencentes ao projeto de desenvolvimento de *software* ou por profissionais de uma área especialmente criada para exercer as funções de teste para todos os projetos de desenvolvimento do *software*.
- Externo: pertencente a uma empresa externa e formado por profissionais especialistas em testes.

O importante é que o grupo de teste se baseie em uma metodologia formal de teste e cumpra – um método que distinga claramente quais são as funções do ciclo de desenvolvimento e as do ciclo de teste.

A experiência tem demonstrado que se alcançam bons resultados quando os testes são realizados pelos profissionais do próprio grupo de desenvolvimento de um projeto, uma vez que é muito difícil fazer com que essas pessoas atuem ao mesmo tempo como desenvolvedores e testadores, com a adoção de metodologias diferentes para cada função (já se constatou que os desenvolvedores tendem a encobrir seus próprios enganos, de modo que a eficácia dos testes dependeria da

capacidade dessas pessoas de atuar com duas metodologias diferentes, de maneira isenta e independente).

6.2 Definição da População-Alvo/Amostra ou Unidade de Análise

Para a realização desta pesquisa será utilizado uma empresa de pequeno porte localizada em Porto Alegre que atua, basicamente, com prestação de serviços de consultoria e desenvolvimento de sistemas, atualmente trabalham nesta empresa aproximadamente dez colaboradores. A escolha desta empresa foi motivada pois a empresa ainda não possui o processo de testes de *software*.

Os modelos de teste atuais são utilizados em empresas de médio a grande porte, pois possuem um processo complexo em que não é considerada a realidade de uma empresa de pequeno porte.

O planejamento do processo de teste deve fazer parte do planejamento global do sistema culminando em um plano de teste que constitui um dos documentos cruciais no ciclo de vida do desenvolvimento de *software* (HOWDEN, 1987). Nesse plano são estimados recursos e são definidos estratégias, métodos e técnicas de teste, caracterizando-se um critério de aceitação do *software* em desenvolvimento. A falta de tempo e recursos, bem como a de uma metodologia adequada para a realização dos testes, são os principais problemas enfrentados pelas empresas de pequeno porte.

6.3 Técnicas de Coleta de Dados

Esta será uma pesquisa qualitativa que por meio de um Estudo de Caso procurará compreender as características e os benefícios de implantar um processo de teste de *software*, bem como a viabilidade da utilização conjunta das mesmas em uma empresa de pequeno porte.

O objetivo da coleta de dados no estudo de caso é conseguir o máximo de informações possíveis para identificar as necessidades das empresas de pequeno porte.

Como fonte da coleta de dados serão utilizadas entrevistas em profundidade, na qual o pesquisador realizará entrevistas semi-estruturadas, empregando perguntas, que possibilitem interpretar e assimilar a expectativa dos participantes da

pesquisa, neste caso à empresa que terá o processo implantado e seus funcionários.

Outra técnica a ser abordada será a observação dos participantes, de forma aberta, na qual o pesquisador terá a permissão da empresa para observar, para entrevistar e para participar no ambiente de trabalho em estudo.

Inicialmente, os dados serão avaliados separadamente, ou seja, será descrita a situação por meio da observação participante perante o que foi diagnosticado. O próximo passo será a comparação das informações para averiguar os dados coletados nas diferentes entrevistas. A análise combinada das informações viabilizará distinguir e confrontar aspectos importantes. E por fim, identificar os principais motivos que levam aos atuais problemas com os indicadores que estão sendo utilizados para avaliar a qualidade do processo de desenvolvimento de *software* que se adapte a realidade da empresa em estudo.

A coleta será efetuada pelo levantamento de informações bibliográficas, no mercado, as entrevistas com a empresa, para identificar as variáveis necessárias que irão compor a pesquisa. Na análise será feita à análise das informações, a identificação das necessidades da empresa, para que se possa analisar a operacionalidade e funcionalidade do processo. Como resultado será feita à avaliação das informações que coletamos para assim serem feitas às devidas conclusões.

6.4 Técnicas de Análise de Dados

Os dados da pesquisa serão registrados através de anotações de campo e entrevistas.

As entrevistas serão transcritas com o intuito de otimizar a mobilização dos dados e conseqüente formulação dos resultados. Para este projeto de pesquisa pretende-se utilizar questionários e planilha de dados.

Os resultados serão formulados através de uma análise comparativa, utilizando como fontes a entrevista semi-estruturada, o questionário e a análise de documentos.

A figura 4 demonstra o processo que será executado para o cumprimento do trabalho.

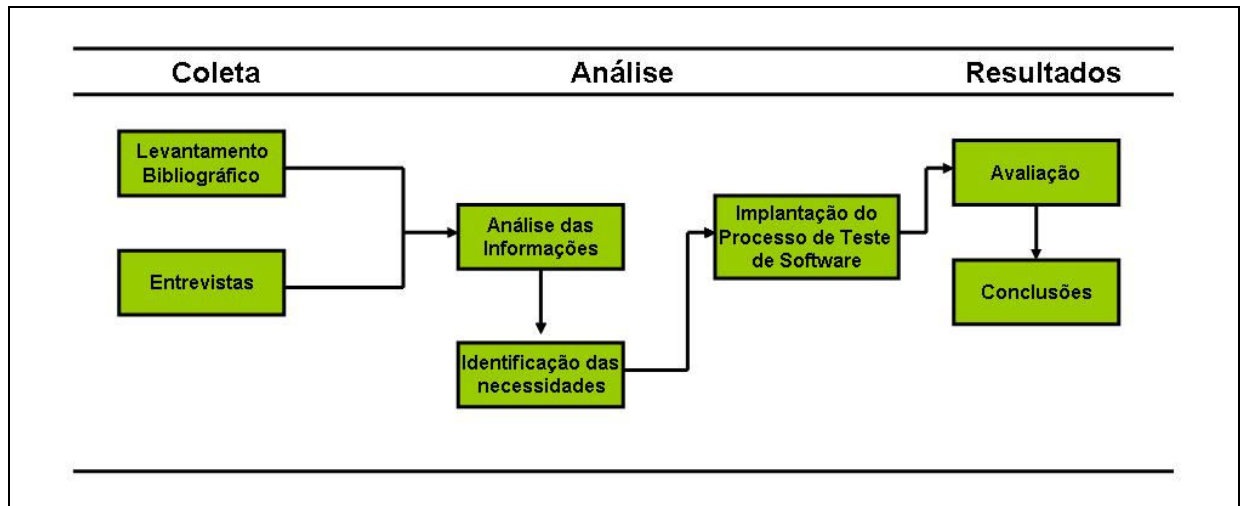


Figura 4 – Processo do Trabalho

Através da tabela 3 é possível visualizar os objetivos específicos e os seus respectivos procedimentos de pesquisa:

Objetivos Específicos	Procedimentos	Resultados	Benefícios
Identificar as necessidades de uma empresa de pequeno porte.	Entrevista.	Obtenção dos dados que são relevantes para o prosseguimento da investigação.	Qualidade
Analisar dos dados relevantes.	Análise de bibliografia, entrevistas e experiência da autora.	Levantamento de informações.	Funcionalidade Confiabilidade Usabilidade Manutenibilidade Portabilidade
Implementar o processo de teste de <i>software</i> na empresa	Análise de bibliografia, entrevistas e experiência da autora.	Recolhimento de referencial teórico para o desenvolvimento da pesquisa	Eficiência Custo
Avaliar o processo de teste de <i>software</i> .	Análise do plano, levando em consideração todas as características do processo de desenvolvimento do <i>software</i> .	Comparar o custo final e a qualidade apresentada em um projeto de desenvolvimento de <i>software</i> antes e após a implantação do processo de teste de <i>software</i> .	Qualidade Custo Tempo

Tabela 3 - Objetivos específicos e procedimentos de pesquisa

7 RESULTADOS

Nessa primeira fase do trabalho (TCCI), foram iniciadas as pesquisas de levantamento de dados que atenderiam as necessidades de uma empresa de pequeno porte. Na segunda fase do trabalho (TCCII) serão refinados essas pesquisas com a sua aplicação em estudo de caso.

A metodologia deste trabalho será realizada em três etapas: levantamento, execução e documentação.

A primeira etapa foi realizada neste trabalho de março até junho de 2009 e foi focado em modelos que poderiam ser adotados por uma empresa de pequeno porte e definido um modelo baseado em bibliografias e empresas que já utilizam essa metodologia.

Algumas alterações foram realizadas na metodologia 3P3E para atender as necessidades de uma empresa de pequeno porte na implantação do processo de teste de software na segunda etapa deste trabalho.

Etapas	Objetivos	Procedimentos	Resultados
Procedimentos Iniciais	Analisar o projeto.	Nesta primeira etapa será realizado um estudo do negócio que dará origem ao sistema a ser desenvolvido.	Através desse estudo será possível identificar as necessidades de recursos, ambiente e tempo para a execução do processo de teste.
Planejamento	Elaborar o plano de execução de teste.	Consiste em elaborar a estratégia de teste e o planos de teste. Estes serão utilizados de modo a minimizar os riscos do negócio e fornecer os caminhos para as próximas etapas.	Esta etapa permanecerá por todo o projeto para garantir que o planejamento seja executado. Caso o escopo do projeto sofra alterações o planejamento de testes poderá sofrer alterações mesmo com o projeto em andamento.
Preparação	Preparar ambiente de teste.	O objetivo desta etapa é preparar o ambiente de teste (equipamento, pessoal, <i>hardware</i> e <i>software</i>).	Garantir que os testes sejam executados em um ambiente semelhante ao de produção e por pessoas qualificadas.
Especificação	Elaboração de Casos de Teste e Planos de Teste.	Serão criados e revisados os casos e roteiros de teste. Esta etapa acompanhará a etapa de desenvolvimento, a medida que a equipe de desenvolvimento liberar alguns módulos ou parte do sistema para teste.	Garantir que os testes sejam executados corretamente. Qualquer alteração na etapa de planejamento o roteiro e casos de teste também deverão ser alterados.
Execução	Realizar os testes.	Executar os testes de acordo com o roteiro e planos criados na etapa anterior. Identificar defeitos, bugs e/ou inconsistências no sistema. Acompanhar as correções e realizar os mesmos testes para verificar sua correção.	Garantir a qualidade do sistema. Ao surgir alterações na etapa de planejamento deverão ser executados testes de regressão, garantindo que a alteração não afetou os outros módulos.
Entrega	Registros de Teste.	Nessa etapa, o projeto de teste é finalizado. Será concluída arquivada toda a sua documentação e serão relatadas todas as ocorrências desse projeto que forem consideradas relevantes à melhoria do processo.	A documentação gerada poderá auxiliar no planejamento de novos projetos.

Tabela 4 – Modelo 3P3E para empresas de pequeno porte

A figura abaixo demonstra o processo de teste utilizando a metodologia 3P3E integrado com o processo de desenvolvimento.

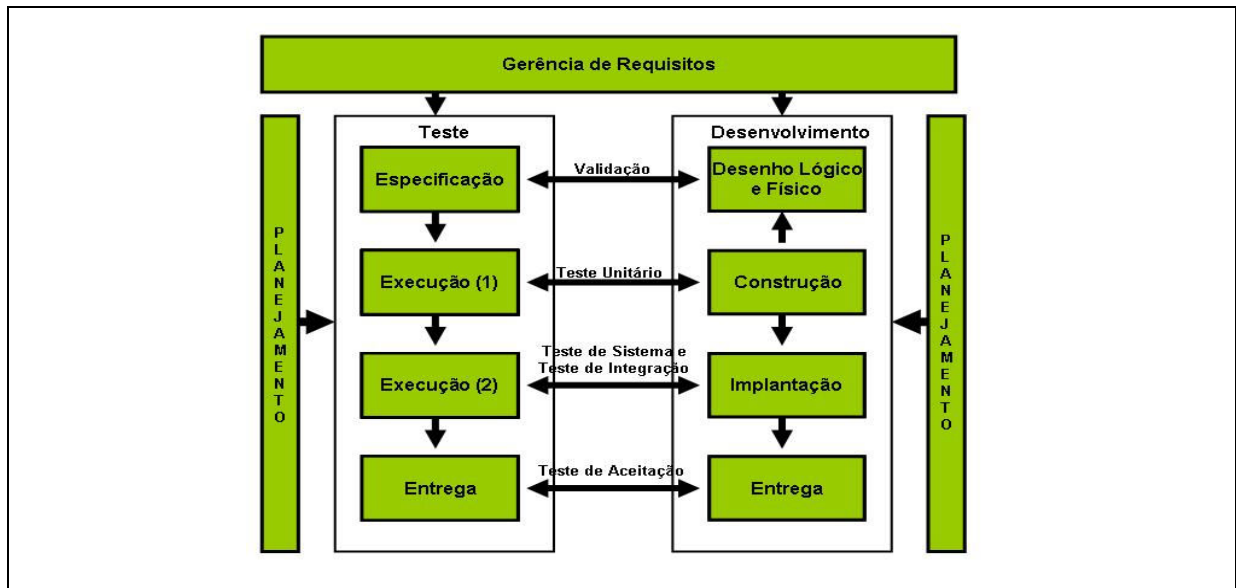


Figura 5 – Processo de teste integrado com o processo de desenvolvimento

A segunda etapa, denominada execução será realizada na segunda fase deste trabalho no período de junho a novembro de 2009, e tem por objetivo aplicar a abordagem para a implantação do processo de teste de *software* com base no modelo definido na primeira fase (levantamento).

A terceira etapa, denominada documentação será realizada em novembro até dezembro de 2009. Nesta etapa será formalizado e documentado o resultado obtido através da implantação do processo de teste em uma empresa de pequeno porte. A geração do resultado será baseada nas respostas de entrevistas realizadas com a equipe envolvida, tornando a avaliação mais confiável, garantindo mais eficácia e maior rapidez no resultado da avaliação.

8 TRABALHO DE CONCLUSÃO DE CURSO - II

No trabalho de conclusão de curso II será utilizada a metodologia abordada no trabalho de conclusão I na implantação de um processo de teste em uma empresa de pequeno porte que não possui esse processo. Será aplicada a metodologia mais adequada a uma empresa de pequeno porte e será avaliado o custo final e a qualidade apresentada em um projeto de desenvolvimento de *software* sem o

processo de teste e um projeto de desenvolvimento de *software* com o processo de teste de *software*.

9 REFERÊNCIAS

- DELAMARO. M.E. ***Mutação de Interface: um critério de adequação interprocedimental para o teste de integração.*** São Carlos, SP, Teste de Doutorado, Instituto de Física de São Carlos, Universidade de São Paulo, jun de 1997.
- HOWDEN, W.E. ***Software Engineering and Technology: Functional Program Testing and Analysis.*** New York: McGrall-Hill Book Co, 1987.
- KRUCHTEN. Philippe. ***The Rational Unified Process – An introduction.*** 2º edição. s.l. Addison Wesley, 2000.
- MARTIN&MCCLURE. ***Técnicas Estruturadas e Case - Makron Books,*** 1984.
- MOLINARI, Leonardo. ***Teste de Software – Produzindo sistemas melhores e mais confiáveis.*** 4º edição. São Paulo – SP. Érica, 2008.
- MYERS. ***The Art of Software Testing.*** 2ª edição. New Jersey: John Wiley & Sons, 2004.
- NBR ISO/IEC 12207:1995. ***Tecnologia da Informação – Processos de ciclo de vida de um software,*** 1997.
- POL, Martin et al. ***Software Testing - A Guide to the TMap Approach - Addison-Wesley,*** 2002.
- PRESSMAN. ***Engenharia de Software.*** 5ª edição. Rio de Janeiro: McGraw-Hill, 2002.
- RIOS, Emerson. ***Bases de conhecimento em teste de software: fonte de consulta a execução de testes de software.*** [et al.] - 2. edição. Rev. São Paulo: Martins, 2007.
- RIOS, Emerson. ***Teste de Software.*** 2º ed. Rio de Janeiro: Alta Books, 2006.
- ROCHA et ali. ***Qualidade de Software Teoria e Prática.*** São Paulo: Prentice Hall, 2001.
- SEBARE. Disponível em <http://www.sebrae.com.br>. Acesso em: junho de 2009.
- SEI. CMMI Product Team: CMMI version 1.1, CMMI-SE/SW/IPPD/SS. Boston: Carnegie Mellon University, 2002.
- STAKE, Robert E. ***Investigación con estudio de casos.*** Madrid: Morata, 1998.
- TRIVIÑOS, Augusto Nivaldo S. ***Bases teórico-metodológicas da pesquisa qualitativa em Ciências Sociais.*** 2.ed. Porto Alegre: Ritter dos Reis, 2001.
- YIN. Robert K. ***Estudo de caso: planejamento e métodos.*** Porto Alegre: Bookman, 2. edição, 2009.