

WESLEY STEVERSON SANTOS DE AZEVEDO

**TESTE DE SOFTWARE: UMA ANÁLISE DOS REQUISITOS FUNCIONAIS DO
PROGRAMA APLICATIVO FISCAL – EMISSOR DE CUPOM FISCAL (PAF-ECF)**

Trabalho de conclusão de curso
apresentado ao Centro Universitário do
Rio Grande do Norte, como requisito final
para obtenção do título de Bacharel em
Sistemas de Informação.

Orientador: Prof. Ytalo Rosendo do
Amaral.

NATAL-RN
2012

Catálogo na Publicação – Biblioteca do UNI-RN
Setor de Processos Técnicos

Azevedo, Wesley Steverson Santos de.

Teste de software: uma análise dos requisitos funcionais do programa aplicativo fiscal – emissor de cupom fiscal (PAF-ECF) / Wesley Steverson Santos de Azevedo. – Natal, 2012.

43f.

Orientador: Prof. Ytalo Rosendo do Amaral.

Monografia (Graduação em Sistemas de Informação) – Centro Universitário do Rio Grande do Norte.

1. Engenharia de Software - Monografia. 2. Software PAC-ECF - Monografia. 3. Aplicativo Fiscal - Monografia. I. Amaral, Ytalo Rosendo do. II. Título.

RN/UNI-RN/BC

CDU 004.41

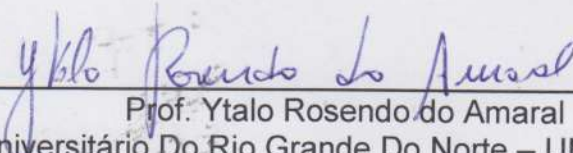
WESLEY STEVERSON SANTOS DE AZEVEDO

**TESTE DOS REQUISITOS FUNCIONAIS DO PROGRAMA APLICATIVO FISCAL –
EMISSOR DE CUPOM FISCAL (PAF-ECF)**

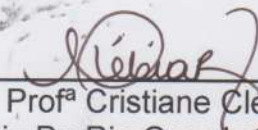
Trabalho de conclusão de curso
apresentado ao Centro Universitário do
Rio Grande do Norte, como requisito final
para obtenção do título de Bacharel em
Sistemas de Informação.

Aprovado em Dez de 2012.

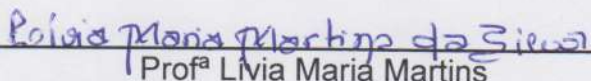
BANCA EXAMINADORA


Prof. Ytalo Rosendo do Amaral

Centro Universitário Do Rio Grande Do Norte – UNI-RN – Natal-RN


Profª Cristiane Clébia Barbosa

Centro Universitário Do Rio Grande Do Norte – UNI-RN – Natal-RN


Profª Livia Maria Martins

Centro Universitário Do Rio Grande Do Norte – UNI-RN – Natal-RN

Natal-RN
2012

Dedico este trabalho a todos que estiveram
comigo nesta jornada. Em especial aos meus
familiares com carinho.

AGRADECIMENTO

Agradeço a Deus pelo dom da vida, pelo seu amor infinito, sabedoria e paciência, a quem devo tudo que sou e o que tenho.

Aos meus pais Francinaldo Rocha e Márcia Rejane, meus maiores exemplos, obrigado pela confiança em mim depositada.

Ao meu irmão Hudson Wagner, por todo amor, carinho e incentivo.

Ao meu orientador, professor Ytalo Rosendo, imensa gratidão.

A professora Cristiane Clébia, pela contribuição e incentivo na minha vida acadêmica, sendo fonte de inspiração e dedicação.

Aos meus amigos de classe, em especial a Clean Cordeiro, a quem aprendi a amar e construir laços eternos. Obrigado por todos os momentos que fomos estudiosos e vagabundos. Obrigado pela mão que sempre foi estendida quando eu precisava.

Obrigado a todos que, mesmo não citados aqui, tanto contribuíram para a conclusão desta etapa.

A vantagem é recíproca, pois os homens,
enquanto ensinam, aprendem.

Sêneca

RESUMO

Realizou-se o levantamento teórico das técnicas de testes, possibilitando a comparação do roteiro de análise funcional de programa aplicativo fiscal – emissor de cupom fiscal – (PAF-ECF) com as respostas do software desenvolvido pela empresa. Nas últimas décadas a Engenharia de Software teve uma evolução expressiva, procurando determinar técnicas, processos, métodos e ferramentas para a construção de softwares, em virtude do crescente uso de sistemas baseados na computação em grande parte das áreas de atuação humana, o que exige uma demanda por qualidade e produtividade, tanto para quem os produzem como para quem os utiliza. Este trabalho foi realizado à luz dos teóricos: Roger Pressman, Ian Sommerville e Emerson Rios. Propõe-se um entendimento sobre as técnicas dos testes de softwares, a importância do teste e expõe os resultados do roteiro de teste para homologação do software junto com o órgão avaliador competente.

Palavras-Chaves: Engenharia de Software, Teste de software, PAF-ECF.

ABSTRACT

We carried out the theoretical survey of testing techniques, enabling the comparison of functional analysis script of tax application program - tax coupon emitter – (PAF-ECF) with the responses of the software developed by the company. In recent decades the Software Engineering had a significant evolution, determining techniques, processes, methods and tools for building software due to the increasing use of computing systems based on the wide areas of human activity, which demands quality and productivity, such for who produces them as for who uses them. This work was accomplished based on the concepts of Roger Pressman, Ian Sommerville and Emerson Rios. We propose the understanding of software testing techniques, the importance of testing and present the test script results for ratification of the software along with the proper appraiser organ.

Keywords: Software Engineering, Software test, PAF-ECF

LISTA DE FIGURAS

Figura 1. Atividades do processo de software.....	16
Figura 2. Ciclo de vida do modelo Cascata.....	17
Figura 3. Ciclo de vida do modelo de Prototipação.....	19
Figura 4. Ciclo de vida do modelo Espiral.....	20
Figura 5. Modelo de qualidade para qualidade externa e interna.....	21
Figura 6. Modelo Tradicional do Custo da Qualidade.....	23
Figura 7. Matriz GUT.....	24
Figura 8. Relação entre níveis, tipos e técnicas de teste.....	25
Figura 9. Estratégia de teste.....	26
Figura 10. Características de um software testável.....	28
Figura 11. Tabela com prazos de obrigatoriedade do PAF-ECF.....	31

LISTA DE ABREVIATURAS E SIGLAS

ICMS	-	Sobre Circulação de Mercadorias e Prestação de Serviços Imposto
ASSESPRO	-	Associação das Empresas Brasileiras de Telecomunicação da Informação
AFRAC	-	Associação Brasileira de Automação Comercial
ISO/IEC	-	Organização Internacional para Padronização / Processo de Desenvolvimento de Software
PC	-	Computadores Pessoais
RICMS/RN	-	Regulamento do ICMS no Rio Grande do Norte
RUDFTO	-	Registro de Utilização de Documentos Fiscais e Termos de Ocorrências
COTEPE	-	Comissão Técnica Permanente
CONFAZ	-	Conselho Nacional de Política Fazendária

SUMÁRIO

1 INTRODUÇÃO	13
1.1 TEMA.....	13
1.2 PROBLEMA.....	13
1.3 OBJETIVOS.....	14
1.3.1 Objetivo Geral.....	14
1.3.2 Objetivo Específico	14
1.4 JUSTIFICATIVA.....	15
2 REFERENCIAL TEÓRICO.....	16
2.1 ENGENHARIA DE SOFTWARE.....	16
2.1.1 Processos de software	16
2.1.1.1 O modelo Cascata	17
2.1.1.2 Modelo de Prototipação.....	18
2.1.1.3 Modelo de Espiral.....	19
2.2 QUALIDADE DE SOFTWARE.....	20
2.3 TESTE DE SOFTWARE	21
2.4 IMPORTÂNCIA DOS TESTES	22
2.5 CUSTO DOS TESTES.....	23
2.6 ESCOLHA DO CONJUNTO DE TESTES.....	24
2.7 DIMENSÕES DO TESTE	25
2.7.1 Estratégia de teste de software.....	26
2.7.1.1 Teste de unidade	26
2.7.1.2 Teste de integração	27
2.7.1.3 Teste de sistema	27
2.7.1.4 Teste de aceitação	27
2.7.2 Fundamentos dos testes de software	27
2.7.2.1 Técnica de teste de software Caixa-Branca	28
2.7.2.2 Técnica de teste de software Caixa-Preta	29
2.7.2.3 Tipos de teste de software.....	29
2.8 PAF-ECF	30
2.8.1 Definição	30
2.8.2 Obrigatoriedade.....	31
2.8.2.1 ECF	31

2.8.2.1 PAF	31
2.8.3 Credenciamento do PAF-ECF	32
2.7.4 Requisitos funcionais do PAF-ECF	33
3 METODOLOGIA	34
4 ANÁLISE DOS RESULTADOS	35
5 CONCLUSÃO	37
5.1 SÍNTESE DO PROJETO	37
5.2 CONTRIBUIÇÕES	37
5.3 LIMITAÇÕES	38
REFERÊNCIAS	39
ANEXO 1 – ROTEIRO DE ANÁLISE FUNCIONAL DE PROGRAMA APLICATIVO FISCAL - EMISSOR DE CUPOM FISCAL PAF-ECF (VERSÃO 1.7)	41

1 INTRODUÇÃO

1.1 TEMA

O Programa Aplicativo Fiscal - PAF é um software devidamente homologado pelo órgão competente, instalado no computador conectado ao Emissor de Cupom Fiscal - ECF. O PAF faz a comunicação com o ECF possibilitando o registro de operações de venda varejo no estabelecimento comercial.

Desde o convênio ICMS 85/01 existe regras para o software que faz a interface com o emissor de cupom fiscal, havia porém critérios e procedimentos diferentes em vários estados. Diante desta realidade, era de se esperar um melhor controle do Fisco¹ sobre o PAF-ECF, havendo assim a necessidade de uniformizar esses critérios, tornando-os iguais em todas as unidades federativas do Brasil. Com isso o Fisco iniciou uma série de contatos com desenvolvedoras, principalmente a ASSESPRO e a AFRAC para que fossem desenvolvidos os requisitos do PAF-ECF. Esses contatos originaram o Ato Cotepe 06/08 e o Convênio ICMS 15/08, tais documentos contém as informações para automação comercial do PAF-ECF que todas as *software-houses* deverão atendê-los.

A primeira obrigatoriedade do uso de tal aplicativo aqui no Rio Grande no Norte se deu em 31/07/2011, a princípio para as empresas que já se encontravam com a obrigatoriedade do uso da ECF.

Testar software é a atividade de comparar o que um item de teste faz com o que é esperado que faça (ISO/IEC-29119).

Neste trabalho será apresentada a análise dos requisitos funcionais do PAF-ECF, no qual contemplará as principais etapas do desenvolvimento de um software, enfatizando a etapa teste, tema deste trabalho.

1.2 PROBLEMA

É evidente que desenvolvedores de softwares em geral sabem que testar é importante, que devem dedicar parte do seu tempo para essa atividade, mas para

¹ Estado enquanto gestor do Tesouro público, no que diz respeito a questões relacionadas com atividades financeiras, tributárias, econômicas.

muitos o teste é visto como uma atividade secundária, a situação mais comum é aquela na qual o teste é realizado junto ao cliente, para provar que o produto funciona, podendo nesse processo serem encontrados inúmeros defeitos, acarretando insegurança e no aumento de custos com a correção dos erros após a entrega do produto final.

O processo de teste é complicado, é impossível testar um software por completo. Contudo, através da engenharia de software em conjunto com uma metodologia de teste, é possível executar com eficiência a atividade de teste. Dessa forma é possível encontrar o maior número de *bugs*² antes da entrega ao usuário final. Garantindo a satisfação do cliente e a redução de custos.

É imprescindível a utilização de um processo formal para o desenvolvimento de um software, com ele fica mais fácil à comunicação entre o setor de teste e as demais áreas da equipe de desenvolvimento.

A empresa em questão realizou técnicas de testes de softwares sob a luz da teoria de Engenharia de Software?

1.3 OBJETIVOS

1.3.1 Objetivo Geral

O objetivo desse estudo é levantar conhecimento de Engenharia de Software para que possa verificar se a análise dos testes funcionais efetuados do software com as regras do Programa de Aplicativo Fiscal, que faz interface com a Emissora de Cupom Fiscal está de acordo com o que foi estudado.

1.3.2 Objetivo Específico

Realizar um estudo bibliográfico da base teórica referente à Engenharia de Software;

Estudar as metodologias e os métodos das técnicas de teste de um software;

Apontar as vantagens e problemas encontrados no teste do software.

² Defeito (*bug*) é um erro no funcionamento comum de um software.

1.4 JUSTIFICATIVA

O grande desafio das empresas é desenvolver softwares de qualidade, em um curto espaço de tempo com o menor custo possível. Portanto, quanto mais cedo for encontrado um defeito, mais barato será sua correção. Partindo desse princípio a atividade de teste é de fundamental importância, tem por finalidade agregar valor ao produto e a redução de custo.

Com uma boa metodologia de teste, o software será mais confiável, um dos principais objetivos de um desenvolvedor.

2 REFERENCIAL TEÓRICO

2.1 ENGENHARIA DE SOFTWARE

Segundo Bauer *apud* Parreira (1998, p. 9), na primeira grande conferencia dedicada ao assunto, define que Engenharia de Software é: "O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que funcione eficientemente com máquinas reais."

Dessa forma, a Engenharia de Software visa à construção de um software que atenda as necessidades dos clientes e que seja cumprido o planejado nas questões prazo e orçamento.

2.1.1 Processos de software

Sommerville (2003, p. 60) define que: "Um processo de software é um conjunto de atividades e resultados associados que levam à produção de um produto de software.". Seja qual for o processo de desenvolvimento de um software, ele obrigatoriamente deverá conter as atividades de: especificação, desenvolvimento, testes e evolução, assim como mostra a figura 1.

Figura 1. Atividades do processo de software.



Fonte: Adaptado de Carneiro, Muto, Colares (2003, p.1).

Existem outros processos no desenvolvimento de um software que serão propostos por alguns modelos apresentados a seguir, estes modelos estabelecem como e em que ordem as atividades deverão ser executadas, proporcionando assim a diminuição dos problemas que poderão surgir no desenvolvimento do software.

2.1.1.1 O modelo Cascata

Segundo Pressman (1995) o modelo em cascata assim como ilustrado na figura 2, é o ciclo de vida mais antigo e mais utilizado na Engenharia de Software, ele contempla as atividades ou ciclos de:

Figura 2. Ciclo de vida do modelo Cascata.



Fonte: Adaptado de Pressman (1995, p. 33).

Engenharia: Fase em que é realizado o levantamento dos dados em nível de sistema, com a intenção de diminuir reparos futuros no mesmo. Conhecido também como estudo de viabilidade.

Análise: Etapa onde são levantados os requisitos funcionais do projeto. Deverão ser documentados e revistos com clientes antes da execução do projeto.

Projeto: Ciclo que consiste em quatro passos distintos do programa, estrutura de dados, arquitetura de software, detalhes procedimentais e caracterização de interface.

Codificação: Momento no qual é implementado o projeto.

Testes: Etapa em que todo o programa é testado, a fim de encontrar erros. Com isso se tem a redução de gastos na correção de erros futuros após a entrega do software ao usuário.

Implantação: Fase em que o sistema é implantado no cliente, podendo ser atualizado em tempos e tempos, com o intuito de realizar a manutenção para a melhoria dos processos; tais manutenções podem ser corretivas, quando o intuito é a correção de erro; adaptativa, no qual a intenção é adaptar às mudanças do ambiente e evolutiva, que se caracteriza pela adição de novas funcionalidades ao software.

2.1.1.2 Modelo de Prototipação

Pressman (1995) afirma que é o ciclo de vida em que o programador desenvolve um protótipo do software que foi encomendado pelo cliente, tal protótipo é utilizado com objetivo de identificar os requisitos do projeto, avaliar sua finalidade, levantar erros e omissões, para que se possa corrigir e ajustar.

Esse modelo é baseado no *feedback*³ dos usuários, podendo o protótipo receber mudanças que alterem formatos, ou mudanças estruturais no mesmo, dependendo do comentário de cada usuário. A figura 3 ilustra como os ciclos de vida são posicionados neste modelo.

³ Palavra inglesa que significa realimentar ou dar resposta a um determinado pedido ou acontecimento.

Figura 3. Ciclo de vida do modelo de Prototipação



Fonte: Adaptado de Pressman (1995, p. 36).

O protótipo pode ser realizado de uma dessas três formas apresentadas:

1. Um protótipo em papel ou em PC, que retrata a interação homem máquina, podendo ver quantas interações haverá.
2. Um protótipo que implemente funções específicas, da função exigida pelo software.
3. Um programa existente que executa parte ou toda a função desejada, mas que tem características que deverão ser melhoradas.

2.1.1.3 Modelo de Espiral

Pressman (1995) define como modelo que contempla as melhores características do ciclo de vida clássico e da prototipação. Do modelo em cascata utiliza-se os aspectos controlados e sistemáticos, já o de prototipação é utilizado para possibilitar que o desenvolvedor aplique tal abordagem em qualquer etapa da evolução do produto, possibilitando a ele simular a versão final do software. Além disso, é acrescentado a esse modelo a análise de risco, o que faltava nos demais paradigmas.

Tal modelo é definido em quatro importantes atividades, são elas:

1. Planejamento: etapa no qual é determinado os objetivos, alternativas e as restrições do ciclo, para isso é considerado os resultados do ciclo anterior ou da análise dos requisitos;

2. Análise dos riscos: momento em que são analisadas as alternativas e resoluções dos erros;
3. Engenharia, Desenvolvimento e verificação da solução escolhida;
4. Avaliação do cliente: etapa em que ocorre a avaliação dos resultados e se é planejado o ciclo seguinte.

Para melhor entendimento desse modelo, ver figura 4 – Ciclo de vida do modelo Espiral.

Figura 4. Ciclo de vida do modelo Espiral.



Fonte: Adaptado de Pressman (1995, p. 39).

2.2 QUALIDADE DE SOFTWARE

Pressman (2006, p.33), define que: qualidade de software é “conformidade com requisitos funcionais e de desempenho explicitamente declarados, normas de desenvolvimento explicitamente documentadas e características implícitas, que são esperadas em todo software desenvolvido profissionalmente”.

Então, qualidade consiste em desenvolver um software que atenda completamente as necessidades dos clientes. Para garantir essa qualidade pode-se utilizar algumas atividades, são elas:

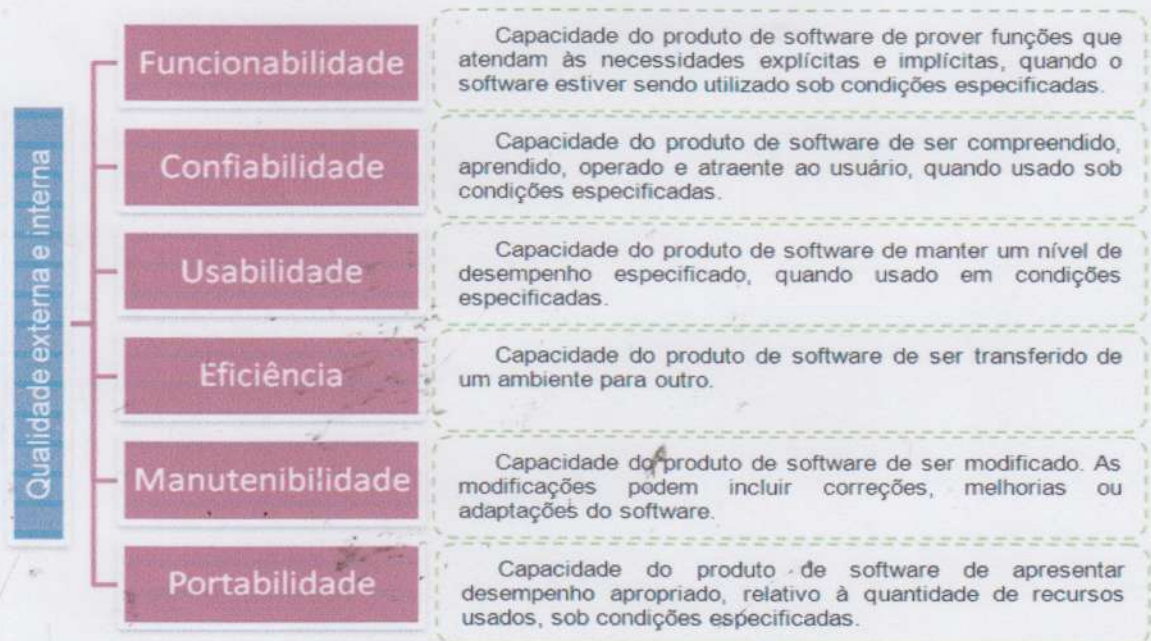
- Revisões de software;
- Testes;
- Padrões e procedimentos formais;
- Controle de mudanças;

- Métricas de software;
- Procedimentos para coleção e disseminação de informações.

Entre as atividades citadas, destaca-se cada vez mais a atividade teste no intuito de detectar defeitos, porém executando os testes juntos com as inspeções, os erros podem ser corrigidos nas fases iniciais do desenvolvimento, dando assim uma visão mais ampla da complexidade do sistema aos desenvolvedores, preparando-os melhor ao confrontar com defeitos mais complexos que poderão surgir durante os testes.

A norma ISO/IEC9126-1 categoriza os atributos de qualidade do software nas seguintes características que a figura 5 ilustra.

Figura 5. Modelo de qualidade para qualidade externa e interna.



Fonte: Adaptado de ISO/IEC9126-1.

2.3 TESTE DE SOFTWARE

Existem inúmeras definições para teste de software, destacamos as seguintes:

- Testar software é a atividade de comparar o que um item de teste faz com o que é esperado que faça. ISO/IEC-29119 *apud* Rios (2012).
- Teste de software é um processo, ou um grupo de processos, definidos para garantir que um código faz o que ele foi desenhado para fazer, e não faz nada que não foi especificado para fazer. Myers *apud* Rios (2012).

- Verificar se o software é executado de forma controlada e está fazendo o que deveria fazer, de acordo com os seus requisitos, e não está fazendo o que não deveria fazer. Rios & Moreira *apud* Rios (2012).

No fim, testar é o processo que impõe a execução de um software, com o objetivo de descobrir falhas, a fim de corrigir antes da entrega do produto final ao cliente. O teste tem uma natureza destrutiva, com o intuito de elevar a confiança do mesmo, tendo em vista que todos os possíveis problemas serão corrigidos antes da entrega final do software.

Os profissionais de testes de software têm como objetivo principal que o software seja desenvolvido e chegue ao cliente próximo da perfeição, com isso, conseguem reduzir significativamente os custos do mesmo.

2.4 IMPORTÂNCIA DOS TESTES

O teste contribui para a qualidade do software, com o teste é possível entregar o software garantindo que o sistema não possui erros críticos, os erros encontrados pela equipe de teste, possuem um custo melhor para a sua correção, o teste possui um papel fundamental na redução dos custos com a manutenção e correção do produto. (LOPES E CARNEIRO, 2010).

Patton *apud* Kammers (2001, p.15) relata que “Os custos para a correção de um problema aumentam exponencialmente no decorrer do projeto”. Portanto, quanto mais tarde encontrado um erro, maior a chance de não existir uma solução viável, com pouco impacto e que se adapte ao tempo disponível.

Atualmente, grande parte dos gastos no ciclo de vida de um software se encontram na manutenção dos mesmos. É evidente que os testes não acabarão com todos os erros, mas aumentam a possibilidade de encontrar muitos deles ainda na etapa de desenvolvimento, diminuindo os transtornos e gastos, e possibilitando a entrega do software com uma menor quantidade de erros. (TOLEDO, 2002).

2.5 CUSTO DOS TESTES

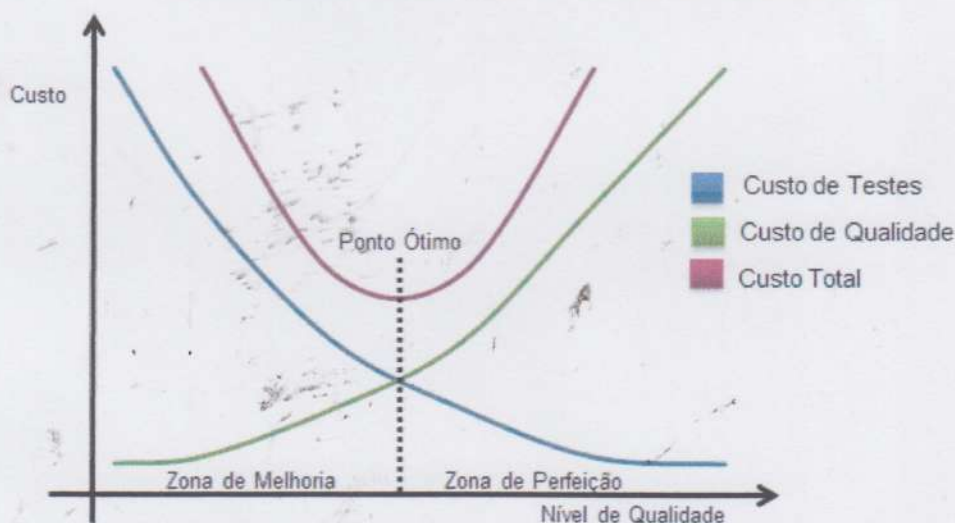
Ficou claro que o teste contribui para a diminuição dos custos diretos, tais como o retrabalho, manutenção e suporte. E nos custos indiretos, satisfação do cliente e qualidade do software.

Pensando no máximo de retorno possível, a pergunta a ser feita é: Quanto devemos investir no teste?

É de conhecimento que apesar do sistema ser testado inúmeras vezes ainda estão sujeito ao surgimento de uma falha. Porém quanto maior investimento no teste, menor será o custo utilizado para a manutenção.

Para melhor esclarecimento utilizaremos a figura 6 para ilustrar o surgimento do custo total, representando a soma do investimento feito nos testes com aqueles que serão utilizados na manutenção e correção.

Figura 6. Modelo Tradicional do Custo da Qualidade.



Fonte: Adaptado de Juran & Gryna *apud* Toledo (2002, p. 10).

O gráfico mostra que: quando os custos de qualidade forem zero, o software será 100% defeituoso e o custo de teste tende a ser muito elevado. Por outro lado, quando o software é 100% dentro da qualidade de conformação, não há falhas, porém os custos de teste tende a ser muito baixo. Jura & Gryna *apud* Toledo (2002).

O ideal é que cada empresa encontre o seu ponto ótimo de investimento em qualidade de software.

2.6 ESCOLHA DO CONJUNTO DE TESTES

Decidir quais os testes serão realizados, é a tarefa mais difícil durante o planejamento. Isso ocorre em virtude da grande quantidade de tipos de testes e variedade de configurações de ambiente, o que pesa na decisão é o tempo de execução dos testes, que poderá ser custosa, ou até extrapolar o limite de entrega do produto.

Kammers (2008, p. 16) sugere que: “podemos utilizar como fator de decisão, uma análise de riscos com seus respectivos impactos, relacionados a cada teste levantado.” E sugere para auxiliar nessa escolha a ferramenta de qualidade *GUT*⁴ como ilustra a figura 7 para determinar quais testes serão executados, e quais não serão.

Figura 7. Matriz GUT.

Indicador	Descrição	Perguntas e pesos
Gravidade	Aquilo que afeta profundamente o resultado. É o dano causado caso o teste não seja executado	• O dano é extremamente importante? (5) • O dano é muito importante? (4) • O dano é importante? (3) • O dano é relativamente importante? (2) • O dano é pouco importante? (1)
Urgência	Pressão do tempo que o sistema sofre. Define qual a importância de se fazer este teste no momento atual.	• Tenho que tomar uma ação muito urgente? (5) • Tenho que tomar uma ação urgente? (4) • Tenho que tomar uma ação relativamente urgente? (3) • Posso aguardar? (2) • Não há pressa? (1)
Tendência	É o estado que a situação pode apresentar, caso não seja tomada nenhuma medida corretiva. Indica a tendência do erro se agravar com o passar do tempo, caso ele ocorra.	• Se não fizer nada, a situação vai piorar muito? (5) • Se não fizer nada, a situação vai piorar? (4) • Se não fizer nada, a situação vai permanecer? (3) • Se não fizer nada, a situação vai melhorar? (2) • Se não fizer nada, a situação vai melhorar muito? (1)

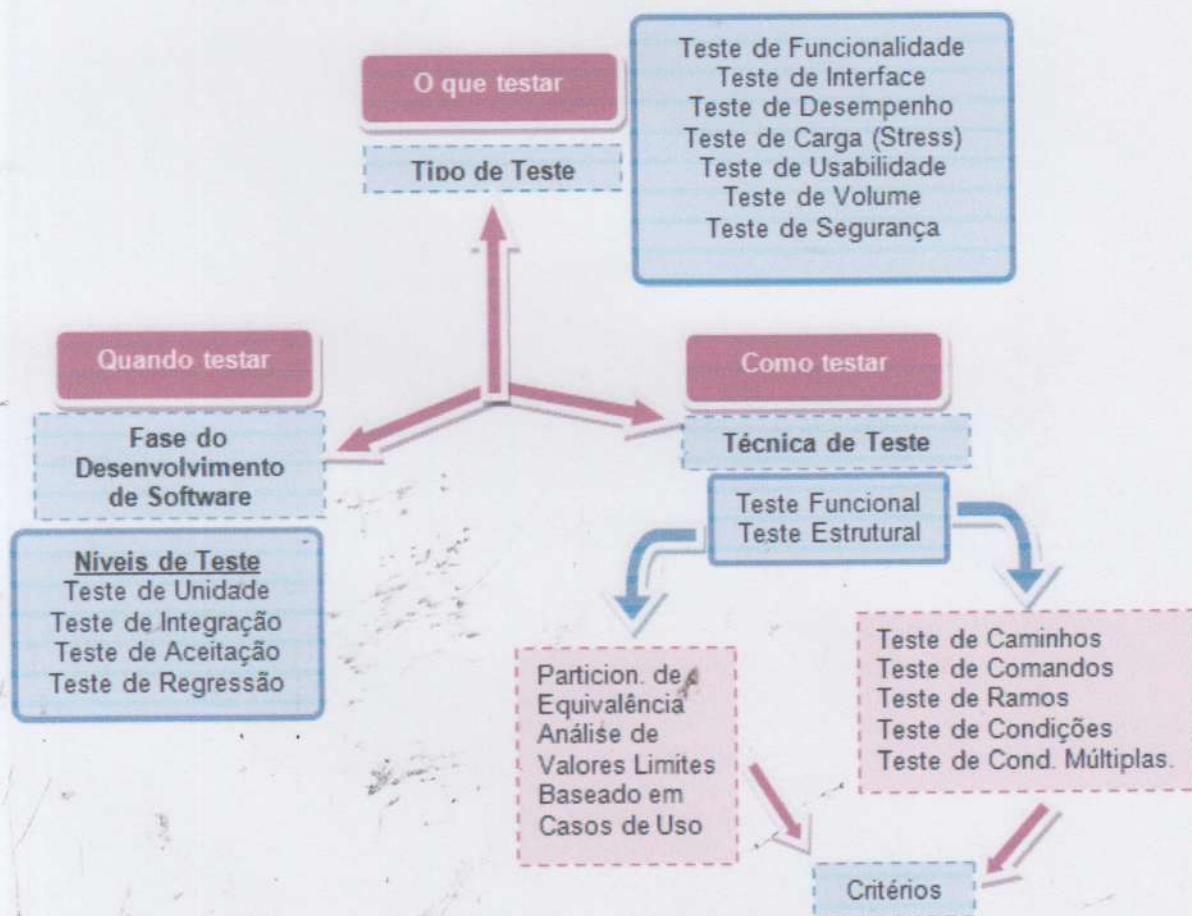
Fonte: Adaptado de Kammers (2008, p.17).

⁴ Matriz GUT é uma forma de se tratar problemas com o objetivo de priorizá-los. Leva em conta a gravidade, a urgência e a tendência, de cada problema.

2.7 DIMENSÕES DO TESTE

Crespo et al.(2004) oferecem uma figura 8 para melhor entendimento sobre as diversas fases e tipos de testes, mostrando a relação existente entre os níveis, as técnicas, os tipos e os critérios de teste que podem ser adotados ao se definir uma estratégia de teste.

Figura 8. Relação entre níveis, tipos e técnicas de teste.



Fonte: Adaptado de Crespo, et al. (2004 p.4).

Onde:

- Fase de desenvolvimento – relaciona o teste com a fase de desenvolvimento onde ele é aplicado.
- Tipo de teste – relaciona o teste à característica de qualidade que está sendo avaliada.
- Técnica de teste – são as técnicas utilizadas para se executar o teste.

2.7.1 Estratégia de teste de software

Segundo Pressman (1995, p.839), “o processo de engenharia de software pode ser visto como uma espiral. A engenharia de sistemas define o papel do software e leva à análise de requisitos de software, movimentando para dentro do centro chegando ao projeto e finalmente à codificação”, Essa visão é ilustrada na figura 9.

A estratégia de software segundo Pressman(1995), também pode ser imaginada como uma espiral, de dentro para fora, onde: relacionando o teste unitário com o que é implementado no código-fonte, o teste de integração com o projeto, o teste de validação com levantamento de requisitos e por fim o teste de sistemas, onde segundo ele “o software e outros elementos do sistema são testados como um todo.” essa visão também é ilustrada na figura 9.

Figura 9. Estratégia de teste.



Fonte: Adaptado de Pressman, (1995, p.841).

2.7.1.1 Teste de unidade

Pressman(1995, p.844), afirma que: “O teste de unidades concentra-se no esforço de verificação da menor unidade de projeto de software – o módulo”, baseando nas técnicas de teste de caixa branca, é aplicado de maneira individual a cada unidade do sistema, e habitualmente é realizado pelo próprio programador.

2.7.1.2 Teste de integração

Para Pressman(1995, p.849) o teste de integração é “uma técnica sistemática para a construção da estrutura de programa, (...) com objetivo de a partir dos módulos testados no nível de unidade, construir a estrutura de programa que foi determinada pelo projeto.” Com o intuito de verificar a interação entre os componentes, é dado mais ênfase á interface.

2.7.1.3 Teste de sistema

Pressman(1995, p.862), define teste de sistema como: “uma série de diferentes testes, cujo propósito primordial é por completamente à prova o sistema baseado em computador.”, baseado nas técnicas de testes de caixa preta, é verificado a conformidade com os requisitos, em local que simule o ambiente de produção.

2.7.1.4 Teste de aceitação

“Quando um software customizado é construído para um cliente, uma série de testes de aceitação é realizada para capacitar o cliente a validar todos os requisitos.” (PRESSMAN, 1995, p.860). Portanto os testes são realizados pelos clientes, podendo está presente no momento do teste um programador, (*teste Alfa*⁵) ou não (*teste Beta*⁶), por esse motivo ele também é conhecido como teste Alfa e Beta.

2.7.2 Fundamentos dos testes de software

Pressman (2006, p. 316), reforça que: “o objetivo de teste é encontrar erros e que um bom teste é aquele que tem alta probabilidade de encontrar erros.”. Com isso é necessário que um engenheiro de software deve projetar e desenvolver um sistema pensando na *testabilidade*⁷.

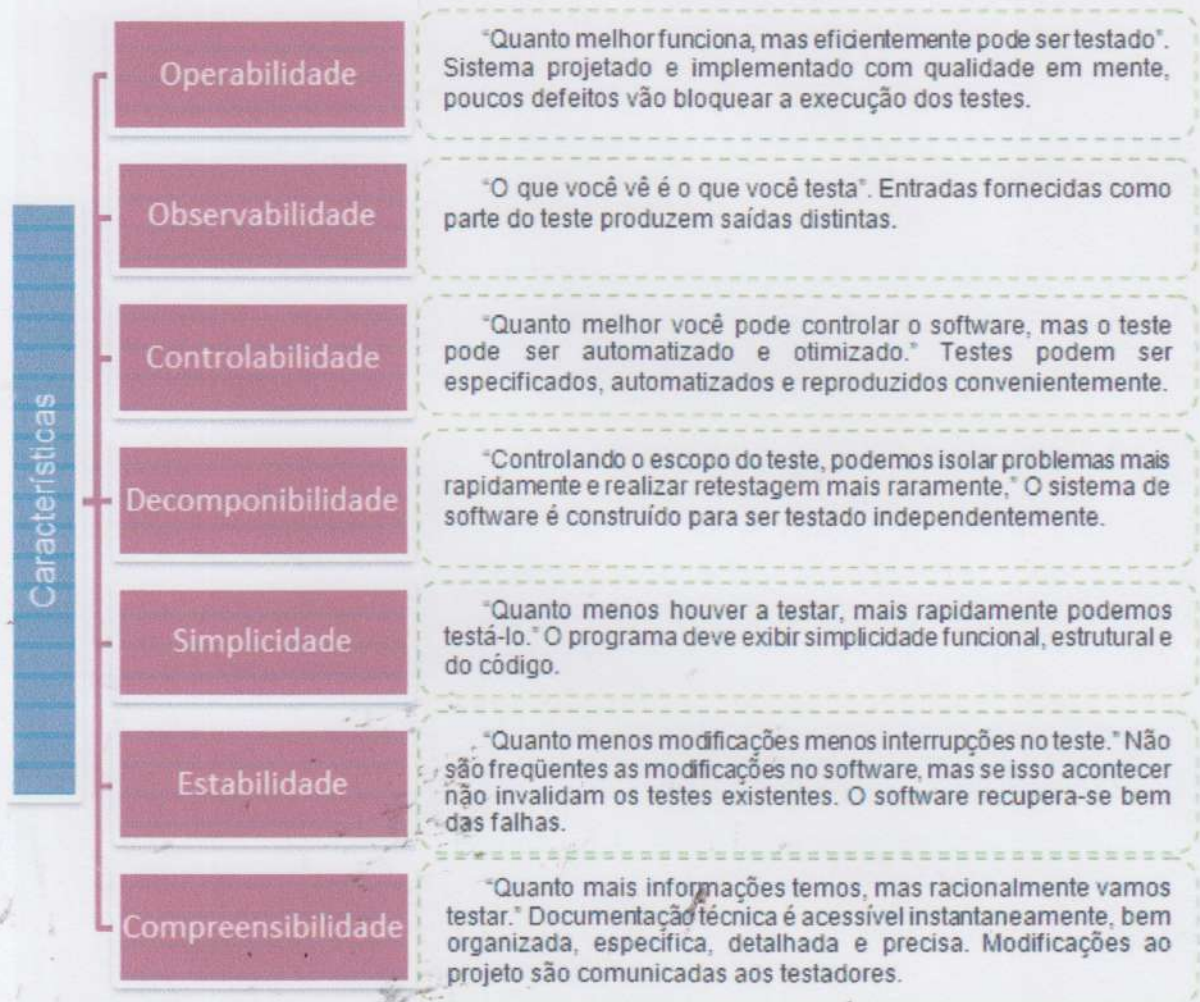
⁵ Teste de aceitação realizado pelo cliente nas instalações do desenvolvedor.

⁶ Teste de aceitação realizado pelo cliente em suas próprias instalações, normalmente sem a presença do desenvolvedor.

⁷ É simplesmente quão fácil um software pode ser testado.

Para que um software possa ser testado é necessário levar em consideração as seguintes características, ver figura 10 para melhor entendimento.

Figura 10. Características de um software testável.



Fonte: Adaptado de Pressman (2006, p. 316).

2.7.2.1 Técnica de teste de software Caixa-Branca

Teste Caixa-Branca, também conhecido como teste estrutural é uma filosofia de projeto de casos de teste que usa a estrutura de controle descrita como parte do projeto ao nível de componentes para derivar casos de teste. (PRESSMAN, 2006).

Então, os testes são realizados pelos próprios desenvolvedores, com a visualização do código-fonte do sistema.

Utilizando métodos de teste caixa-branca, pode derivar casos de testes que:

1. Garantam que todos os caminhos independentes de um módulo tenham sido exercitados pelo menos uma vez.
2. Exercitem todas as decisões lógicas em seus lados verdadeiro e falso;
3. Executem todos os ciclos nos seus limites e dentro de seus intervalos operacionais;
4. Exercitem as estruturas de dados internas para garantir sua validade.

2.7.2.2 Técnica de teste de software Caixa-Preta

Teste Caixa-Preta, também conhecido como teste funcional no qual focaliza os requisitos funcionais do software, ele permite ao engenheiro de software derivar conjuntos de condições de entradas que vão exercitar plenamente todos os requisitos funcionais de um programa. Essa técnica vem a complementar a técnica Caixa-Branca. (PRESSMAN, 2006).

Segundo o mesmo, utilizando métodos de teste caixa-branca, pode derivar casos de testes que resolvam problemas de:

1. Funções incorretas ou omitidas;
2. Erros de interface;
3. Erros de estrutura de dados ou de acesso à base de dados externa;
4. Erros de comportamento ou desempenho;
5. Erros de iniciação e término.

2.7.2.3 Tipos de teste de software

Além dos testes vistos no item 2.7.1 *Estratégia de teste de software* deste trabalho existe outros testes que poderão ser utilizadas para atender a norma ISO/IEC 9126-1, a escolha de um específico vai depender do produto que está sendo testado e o ambiente no qual o mesmo será usado, são eles:

Pressman (1995) defende que:

- Teste de recuperação: é um teste de sistema que força o software a falhar de diversas maneiras e verifica se a recuperação é adequadamente executada.
- Teste de segurança: tenta verificar se todos os mecanismos de proteção embutidos num sistema a proteção, de fato, de acessos indevidos.

- Teste de estresse: é realizado para confrontar os programas com situações anormais, executa o sistema de uma forma que exige recursos em quantidade, frequência ou volume anormal. Essencialmente, o analista tenta destruir o programa.

- Teste desempenho: é idealizado para testar o desempenho de *run-time*⁸ do software dentro do contexto de um sistema integrado.

Barbosa & Torres (2011, p. 50) defende que:

- Teste Operacional: Garante que a aplicação se mantém estável por muito tempo sem falhar.

- Teste Positivo-Negativo: Garante que a aplicação vai funcionar no “caminho feliz” de sua execução e vai funcionar no seu fluxo de exceção.

- Teste de Regressão: Toda vez que algo for mudado, deve ser testada toda a aplicação novamente.

- Teste de Interface: Verifica se a navegabilidade e os objetivos da tela funcionam como especificados e se atendem da melhor forma ao usuário.

- Teste de Carga: Verifica o funcionamento da aplicação com a utilização de uma quantidade grande de usuários simultâneos.

- Teste de Aceitação do Usuário: Testa se a solução será bem vista pelo usuário. Ex: caso exista um botão pequeno demais para executar uma função, isso deve ser criticado em fase de testes.

- Teste de Volume: Testar a quantidade de dados envolvidos.

- Testes de Configuração: Testar se a aplicação funciona corretamente em diferentes ambientes de hardware ou de software.

- Testes de Instalação: Testar se a instalação da aplicação obteve sucesso.

2.8 PAF-ECF

2.8.1 Definição

PAF-ECF (Programa Aplicativo Fiscal do Emissor de Cupom Fiscal). Segundo definição do CONVÊNIO ICMS 9, de 3 de abril de 2009, PAF-ECF é o “programa aplicativo desenvolvido para possibilitar o envio de comandos ao Software Básico do

⁸ É o período durante o qual um programa de computador está executando

ECF, sem capacidade de alterá-lo ou ignorá-lo, para utilização pelo contribuinte usuário do ECF”.

2.8.2 Obrigatoriedade

2.8.2.1 ECF

De forma geral, os estabelecimentos que exerçam atividade de venda ou revenda de mercadorias ou bens ou de prestação de serviços em que o adquirente ou tomador seja pessoa física ou jurídica não-contribuinte do ICMS estão obrigados ao uso do ECF.

O art. 782 § 8 do RICMS/RN enfatiza que:

Somente será permitida a emissão de documento fiscal por qualquer outro meio, inclusive o manual, por razões de força maior ou caso fortuito, tais como falta de energia elétrica, quebra ou furto do equipamento, e nas condições previstas no Convênio SINIEF s/n de 15 de dezembro de 1970, devendo o usuário anotar o motivo no livro de Registro de Utilização de Documentos Fiscais e Termos de Ocorrência (RUDFTO).

2.8.2.1 PAF

Todos os estabelecimentos que possuem ECF, mesmo que usem espontaneamente este equipamento, estão obrigados ao uso do PAF-ECF nos moldes exigidos pela legislação tributária estadual.

Para o estado do Rio Grande do Norte, os prazos para uso do PAF-ECF estão discriminados conforme tabelas abaixo, figura 11.

Figura 11. Tabela com prazos de obrigatoriedade do PAF-ECF.

1 – QUE NÃO SEJAM USUÁRIAS DOS MESMOS (EXCETO FILIAIS EXISTENTES OU NOVAS FILIAIS DE EMPRESAS JÁ USUÁRIAS DE ECF):	
Prazo	Quando da autorização do ECF
Até 31/07/2011	Os ECF's poderão ser autorizados com PAF-ECF ou outro aplicativo mediante apresentação de declaração conjunta (como já vem sendo feito).
A partir de 01/08/2011	Os ECF's só poderão ser autorizados juntamente com um PAF-ECF.

2 – QUE JÁ SEJAM USUÁRIAS DOS MESMOS BEM COMO SUAS FILIAIS EXISTENTES OU NOVAS FILIAIS QUE VENHAM A SER CRIADAS:

Prazo	Quando da autorização do ECF
Até 31/12/2011	Os ECF's poderão ser autorizados com PAF-ECF ou outro aplicativo mediante apresentação de declaração conjunta (como já vem sendo feito).
A partir de 01/01/2012	Os ECF's só poderão ser autorizados juntamente com um PAF-ECF.

3 – QUE SEJAM USUÁRIAS DE APLICATIVOS QUE NÃO SEJAM PAF-ECF AUTORIZÁVEIS PELA SET:

Prazo	Quando da autorização do ECF
Até 31/12/2011	Deverão providenciar a troca dos referidos aplicativos por um PAF-ECF autorizável e solicitar autorização de uso do mesmo na SET.

4 – QUE SEJAM USUÁRIAS DE APLICATIVOS QUE SEJAM PAF-ECF AUTORIZÁVEIS PELA SET E QUE NÃO TENHAM SIDO AUTORIZADOS PARA O CONTRIBUINTE:

Prazo	Quando da autorização do ECF
Até 31/12/2011	Deverão solicitar autorização de uso do referido PAF-ECF à SET.

Fonte: Adaptado de Secretária de Estado de Tributação do Rio Grande do Norte.

2.8.3 Credenciamento do PAF-ECF

Segundo a Secretária de Estado de Tributação do Rio Grande do Norte os passos são os seguintes:

- Primeiramente, a empresa deve desenvolver o PAF-ECF conforme as normas do Ato COTEPE 06/08 para depois submetê-lo para análise no órgão técnico credenciado.
- O órgão técnico emitirá laudo da análise funcional do PAF-ECF e com base nesse laudo, caso haja aprovação, a empresa desenvolvedora solicitará ao CONFAZ o despacho comunicando o registro do PAF-ECF.
- Após a publicação do despacho do CONFAZ, a empresa desenvolvedora apresenta o LAUDO e demais documentos na UF onde deseja se credenciar e então, verificada a documentação. A SEFAZ credencia a empresa e o PAF-ECF para uso no Estado.

2.7.4 Requisitos funcionais do PAF-ECF

Os requisitos funcionais do PAF-ECF foram estabelecidos pelo ATOCOTEPE 06/08. Dentre eles se destacam:

O PAF-ECF ou Sistema de Gestão (Retaguarda) não deve possibilitar ao usuário possuir informações contábeis diversas daquela que é, por lei, fornecida à Fazenda Pública, conforme inciso V do Art. 2º da Lei 8.137/90.

O PAF-ECF deve estar integrado aos Sistemas de Gestão (SG) e Retaguarda para viabilizar a utilização, considerando como integração a capacidade de importar e exportar dados reciprocamente.

O PAF-ECF deve ser instalado de forma a possibilitar o funcionamento do ECF independentemente da rede, ainda que eventualmente a utilize, exceto quando destinado à utilização exclusiva para o transporte de passageiros.

A emissão de registro de Pré-Venda e a emissão de DAV (Documento Auxiliar de Venda) possuem definições específicas no Ato COTEPE 06/08.

O Ato COTEPE 06/08 também traz requisitos específicos do PAF-ECF para: estabelecimento revendedor varejista de combustível automotivo;

- Restaurantes, bares e estabelecimentos similares, incluindo controle de "conta de clientes";
- Farmácia de manipulação;
- Oficina de conserto;
- Transporte de passageiros.

3 METODOLOGIA

Trata-se de uma pesquisa experimental onde o objetivo é levantar todas as atividades necessárias para gerenciar um processo de teste. Dessa forma auxiliando na coordenação das tarefas realizadas pela equipe de testes, alinhando assim o processo de teste com o processo de desenvolvimento e verificar se a análise dos testes funcionais efetuados do software estão de acordo com o que foi estudado.

Houve a necessidade de fundamentar os aspectos essenciais envolvidos nessa atividade, para tal foram realizadas pesquisas através de livros e da internet, identificando os conceitos e metodologias conhecidas.

Após isto, foi feito um estudo mais detalhado no roteiro de análise funcional do Programa Aplicativo Fiscal – Emissor de Cupom Fiscal (PAF-ECF), disponibilizado pela Secretaria de Estado de Tributação do Rio Grande do Norte, para entender o quê, e como seria testado.

E finalmente colocado o aplicativo em testes seguindo a bibliografia pesquisada e o roteiro de teste mencionado no parágrafo acima.

O teste requer um planejamento detalhado em toda atividade de desenvolvimento de software. Uma boa documentação com bons levantamentos de requisitos contribuem para um melhor gerenciamento da execução dos testes, tal documento é o elo de comunicação entre o setor de teste com o setor de desenvolvimento.

Tal análise foi realizada seguindo o roteiro de teste, disponibilizado pela Secretaria de Tributação do Rio Grande do Norte, os testes foram executados de forma sequencial de acordo com que era apresentado no roteiro, caso fosse encontrado algum erro, o mesmo era corrigido pelo setor responsável e depois retomado o teste a partir do requisito que foi encontrado o erro.

Além de seguir rigorosamente o roteiro de teste, foi levado em consideração todo o conhecimento adquirido no desenvolver deste trabalho para melhorar quanto às características de: funcionalidade, confiabilidade, usabilidade e eficiência.

Através de todo o esforço exigido nesse trabalho à empresa no qual o software foi testado, obteve o laudo do órgão técnico credenciado pelo Confaz para homologação de programas aplicativos fiscais, laudo esse de análise funcional do PAF-ECF na versão 1.7 aplicável a ER versão 1.10 do Ato Cotepe nº 51.

4 ANÁLISE DOS RESULTADOS

O trabalho se desenvolveu na maior parte do tempo baseado na técnica de teste caixa preta também conhecida como funcional, onde o roteiro de teste foi levado à risca para que se pudesse chegar ao objetivo. Para realizar o teste funcional foi confrontado o que o roteiro de teste solicitava com as resposta que o software emitia, verificando assim se o software estava apto a receber o laudo, para melhor entendimento verificar o Anexo I, que contempla um requisito entre a diversidade de quarenta e dois do roteiro de teste.

Por muitas vezes houve a necessidade de utilização da técnica de caixa branca, conhecida como estrutural, no momento que o teste funcional de um determinado requisito não foi aprovado e existiu a necessidade de correção pelo desenvolvedor, que posteriormente a correção, utilizou-se do teste de unidade para provar que o que foi corrigido realmente estava de acordo com o que deveria ser.

Também pelo programador foi utilizado o teste de sistema, com o propósito de verificar a totalidade do aplicativo desenvolvido.

O teste de aceitação nesse projeto foi simulado pela equipe de teste, com conhecimento prévio de como alguns clientes da empresa em questão atua, foi possível simular um ambiente de produção para validar as regras adicionadas ao software que os mesmos já utilizavam, nesse ambiente simulado pode também ser testados a recuperação, segurança, estresse, desempenho e interface, podendo assim garantir que ao entregar da versão final ao cliente o número de erros encontrados estivesse próximo a zero.

Com este trabalho, podemos notar a complexidade que o processo de teste exige. E que foi de fundamental importância à existência do roteiro de testes. No qual possuem testes bem elaborados, mas que por muitos acabam dificultando a implantação no cliente. Dentre eles podemos citar a seguintes situações:

Em diversos segmentos do comércio é necessário se fazer a reserva do produto no ato da venda, e por esse motivo a maioria dos contribuintes fazem uso de pré-venda¹². De acordo com o requisito 'V', teste 004 do roteiro de análise técnica do PAF-ECF essa opção não tem disponibilidade de impressão e com isso o que tem acontecido na prática é: Se faz a pré-venda e depois se digita tudo em um DAV¹³, com campos importantes para o processo de separação e entrega da mercadoria, além de informações relevantes como, local de entrega, data e hora,

responsável pela entrega, peso da mercadoria, referência do produto, identificação da localização do produto no estoque, entre outras informações. Ocasionalmente assim retrabalho ao ter que digitar os respectivos dados da venda, comitando na insatisfação dos usuários.

Outro caso interessante é que no Requisito 'III', teste 001 caso exista alguma falha na comunicação de rede, impossibilitando assim a conexão com a base de dados central, é permitido que o PAF-ECF trabalhe em modo stand alone, ou seja, sem conexão com a base de dados central, porém ao se recuperar desta instabilidade faz necessário a atualização do estoque na base central, então imagine a situação: Ocorreu um problema, todos os PAF-ECF trabalhando stand alone, quantidade de um produto como exemplo, caixa d'água igual à 1, não é permitido vender sem estoque, um cliente compra uma unidade de caixa d'água de 20L, um outro cliente compra outra unidade em outro caixa, é permitido porque ambos os aplicativos estão com estoque, porém ao recuperar a estabilidade, é realizado o ajuste do estoque e o estoque fica negativo, ou seja, aos olhos do fisco esse registro está irregular, mas não existe a possibilidade do controle, tendo em vista que não é possível passar com uma caixa d'água de 20L no caixa, realizando a devida reserva do estoque.

Para que não houvesse nenhum imprevisto e que não se conseguisse o objetivo final, foi necessário que todos os envolvidos no processo de desenvolvimento do software estivessem alinhados.

Por fim, pode-se concluir que o PAF-ECF é mais uma de muitas exigências fiscais que o governo impõe ao contribuinte, com a finalidade de evitar o máximo de sonegação possível, ou seja, maior retorno através de impostos para o eles. Também se concluiu que seja qual for a técnica de teste utilizada é fundamental a existência de um, sem teste é quase, ou impossível entregar um software de qualidade ao usuário final.

5 CONCLUSÃO

5.1 SÍNTESE DO PROJETO

Através da pesquisa feita em livros e na Internet, foram reforçados os conceitos essenciais para a atividade de teste. E após esta análise podemos evidenciar que a etapa teste, possui alguns pontos a serem melhorados, que é difícil seguir a risca qualquer tipo de metodologia, mas que é essencial para desenvolver um sistema com qualidade, no prazo determinado e com um custo desejável.

Através deste trabalho pôde-se corrigir as falhas melhorando as funcionalidade antes da homologação junto ao órgão competente e da atualização dos clientes.

Realizando um paralelo com o processo de desenvolvimento, constatou-se que os desenvolvedores de software em geral sabem que testar é importante, que devem dedicar parte do seu tempo para a atividade. Entretanto, a situação mais comum é aquela na qual teste adequados acabam sendo realizados somente após a implantação do sistema no cliente, dessa forma podendo desenvolver um sistema com pouca qualidade, custos maiores e entrega fora do prazo para o cliente.

O intuito desse trabalho foi atingido ao conseguir reduzir os custos do desenvolvimento de um software e na entrega de produtos com uma melhor qualidade.

5.2 CONTRIBUIÇÕES

Podemos citar as seguintes contribuições deste trabalho:

- O levantamento da importância da atividade de teste ao se desenvolver um software.
- A elaboração de uma metodologia embasada nos levantamentos realizados através desse trabalho, para que se possa estruturar e coordenar uma equipe de testes.
- O melhoramento da qualidade dos softwares desenvolvidos pela empresa analisada.

5.3 LIMITAÇÕES

O trabalho foi aplicado em um projeto que já estava em funcionamento, porém sem as funcionalidades que o roteiro de análise dos requisitos funcionais do programa que faz interface com o emissor de cupom fiscal exige, para solucionar tal limitação foi necessário que antes a equipe de desenvolvedores programasse tais funcionalidades, e para isso demandou tempo.

Para que uma pessoa possa executar as atividades abordadas nesse trabalho, é necessário ter conhecimento das técnicas de teste e suas aplicações, o que demanda mais tempo para treinar a equipe de teste.

REFERÊNCIAS

BARBOSA, F. B. e TORRES, I. V. **O Teste de Software no Mercado de Trabalho**. Revista Tecnologias em Projeção v.2, n.1, p. 49-52, Jun., 2011. (Artigo). Disponível em: <<http://revista.faculdadeprojecao.edu.br/revista/index.php/projecao2/article/viewFile/82/70>> acesso em: 04/09/2012.

BRASIL. **Convênio ICMS 9, de 3 de Abril de 2009**. Disponível em: <http://www.fazenda.gov.br/confaz/confaz/convenios/icms/2009/CV009_09.htm> acesso em: 01/10/2012.

_____. **Regulamento do ICMS**. Disponível em: <<http://www.set.rn.gov.br/set/leis/regulamentos/ricms.htm>> acesso em: 16/10/2012.

_____. **ATO COTEPE/ICMS N° 6, de 14 de Abril de 2008**. Disponível em: <http://www.fazenda.gov.br/confaz/confaz/atos/atos_cotepe/2008/ac006_08.htm>, acesso em: 01/11/2012.

CARNEIRO, AllanGuerreiro.MUTO. Cleyton, COLARES, Adolfo. **Agenda de Atividades para Processos de Software**. Belém, 2011. (Artigo). Disponível em: <http://www.univicosa.com.br/arquivos_internos/artigos/Allan.pdf> acesso em: 14/09/2012.

CRESPO, Adalberto Nobiato. SILVA, Odair Jacinto da. BORGES, Carlos Alberto. SALVIANO, Clênio Figueiredo. TEIVE, Miguel de. JUNIOR, Argollo. JINO, Mario. **Uma Metodologia para Teste de Software no Contexto da Melhoria de Processo**. Campinas, 2004. Disponível em: <<http://ebookbrowse.com/uma-metodologia-para-teste-de-software-no-contexto-da-melhoria-de-processo-pdf-d88500851>> acesso em: 16/09/2012.

HERBERT, Juliana Silva; PRICE, Ana Maria Alencar. **Técnicas de Teste de Software Orientado a Objetos**. 2000. (Artigo). Disponível em: <<http://dc407.4shared.com/doc/G5ryikyH/preview.html>> acesso em: 11/10/2012.

KAMMERS, Rodrigo A. **O processo de teste e as atividades envolvidas em cada fase**. 2006, 59f. Trabalho de Conclusão de Curso (Engenharia de Software) - Centro Universitário Positivo, CURITIBA. Disponível em: <<http://pt.scribd.com/doc/43353460/Teste-de-Software-Monografia-Rodrigo-Augusto-Kammers>> acesso em: 12/10/2012.

LOPES, Mateus Bruno T.; CARNEIRO, Allan Guerreiro. **A Importância do Processo de Teste de Software em TI**. (Artigo). Disponível em: <http://www.univicosa.com.br/arquivos_internos/artigos/ImportanciadoProcessodeTestedeSoftwareemTI.pdf> acesso em: 19/10/2012.

NBR ISO/IEC 9126-1, **Engenharia de software - Qualidade de produto Parte 1: Modelo de qualidade** ABNT. Rio de Janeiro, 2003. Disponível em:

<<http://ebookbrowse.com/nbr-iso-iec-9126-1-pdf-d152850799>> acesso em: 02/11/2012.

PARREIRA, Walteno Martins. **Engenharia de Software**. Minas Gerais, 1998. (Apostila). Disponível em: <http://www.waltenomartins.com.br/ap_scd_v1.pdf> acesso em: 11/11/2012

PRESSMAN, R.S. **Engenharia de Software**, Pearson Makron Books, São Paulo, 1995.

_____. **Engenharia de Software**. Tradução de Rosângela Dellosso Penteado, Fernão Stella R. Germano. 6 ed. São Paulo: McGraw-Hill, 2006. Título original: Software Engineering: A Practitioner's Approach.

RIOS, Emerson. **O que é teste de software**. Disponível em: <<http://www.iteste.com.br/LinkClick.aspx?fileticket=N7iNane96WY%3d&tabid=249&mid=440>> acesso em: 13/09/2012.

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

TOLEDO, José Carlos. **Conceitos sobre custos da qualidade**. São Carlos, 2002 (Apostila). Disponível em: <<http://www.gepeq.dep.ufscar.br/arquivos/CustosdaQualidadeApostila.pdf>> acesso em: 20/11/2012.

ANEXO 1 – ROTEIRO DE ANÁLISE FUNCIONAL DE PROGRAMA APLICATIVO FISCAL - EMISSOR DE CUPOM FISCAL PAF-ECF (VERSÃO 1.7)

REQUISITO V TESTES

(Aplicáveis no caso de programa aplicativo dotado de função que possibilite o registro de itens somente na tela do monitor sem impressão de documento que descreva os itens registrados, não podendo esta função estar disponível no ponto de venda, exceto quando se tratar de mesclagem).

TESTE 004: Registro de operação de pré-venda e do Cupom Fiscal respectivo.

Passo 1: Registre no programa aplicativo uma operação de pré-venda com pelo menos um item.

Passo 2: Observe se houve a impressão de algum documento que contenha a descrição dos itens registrados na operação de pré-venda.

Passo 3: Efetive a operação de pré-venda acionando o comando de emissão do Cupom Fiscal correspondente à pré-venda.

Passo 4: Verifique se o Cupom Fiscal emitido contém o número da operação de pré-venda em conformidade com o estabelecido no subitem "a" do item 2.1, conforme o modelo de ECF.

Condição para requisito atendido: Não impressão de documento que contenha a descrição dos itens registrados na operação de pré-venda e emissão do Cupom Fiscal respectivo contendo o número da operação de pré-venda na forma estabelecida no item 2.1.

Condição para requisito não atendido: Impressão de documento que contenha a descrição dos itens registrados na operação de pré-venda ou falta de emissão do Cupom Fiscal respectivo ou emissão do Cupom Fiscal respectivo que não contenha o número da operação de pré-venda ou impressão no Cupom Fiscal respectivo do número da pré-venda de forma diversa da estabelecida no item 2.

TESTE 004-A: Registro de operação de duas ou mais pré-vendas mescladas para uma nova pré-venda e do Cupom Fiscal respectivo.

Passo 1: Registre no programa aplicativo duas ou mais operações de pré-venda com pelo menos um item cada.

Passo 2: Observe se houve a impressão de algum documento que contenha a descrição dos itens registrados em cada operação de pré-venda.

Passo 3: Verifique se é possível informar mais do que uma pré-venda por Cupom Fiscal e se for o caso, comande a emissão do Cupom Fiscal.

Condição para requisito atendido: Não impressão de documento que contenha a descrição dos itens registrados nas operações de pré-venda e emissão de um Cupom Fiscal para cada operação de pré-venda na forma estabelecida no item 2.1.

Condição para requisito não atendido: Impressão de documento que contenha a descrição dos itens registrados nas operações de pré-venda ou falta de emissão do Cupom Fiscal respectivo a cada uma das pré-vendas ou emissão do Cupom Fiscal respectivo que não contenha o número da operação de pré-venda ou impressão no Cupom Fiscal respectivo do número da pré-venda de forma diversa da estabelecida

no item 2.1 ou impressão de Cupom Fiscal contendo mais de um número de pré-venda ou itens referentes a mais de uma pré-venda.

Passo 4: Caso a condição anterior tenha sido atendida, execute a função de mesclar as operações de pré-venda registradas no passo 1, dando origem a uma nova pré-venda contendo a totalidade dos itens registrados nas respectivas pré-vendas seguido da automática impressão e cancelamento dos Cupons Fiscais das pré-vendas originais e comande a emissão do Cupom Fiscal correspondente à nova pré-venda.

Passo 5: Verifique se o Cupom Fiscal emitido contém o número da nova operação de pré-venda em conformidade com o estabelecido no subitem "a" do item 2.1, conforme o modelo de ECF.

Condição para requisito atendido: Não impressão de documento que contenha a descrição dos itens registrados na nova operação de pré-venda. Automática impressão e cancelamento dos Cupons Fiscais das pré-vendas originais e emissão do Cupom Fiscal correspondente a nova pré-venda, contendo o número da operação da nova pré-venda na forma estabelecida no item 2.1.

Condição para requisito não atendido: Impressão de documento que contenha a descrição dos itens registrados na operação de pré-venda ou falta de emissão do Cupom Fiscal respectivo ou emissão do Cupom Fiscal respectivo que não contenha o número da operação da nova pré-venda ou impressão no Cupom Fiscal respectivo do número da nova pré-venda de forma diversa da estabelecida no item 2.1 ou não possibilidade da automática impressão e cancelamento dos Cupons Fiscais das pré-vendas originais.

TESTE 005: Registro de operação de pré-venda e seu posterior cancelamento.

Passo 1: Registre no programa aplicativo uma operação de pré-venda com pelo menos um item.

Passo 2: Localize nos menus do programa a opção que possibilita o cancelamento da operação de pré-venda.

Passo 3: Caso exista função de cancelamento, registre o cancelamento da pré-venda.

Passo 4: Observe se o ECF emitiu o Cupom Fiscal contendo os itens registrados na pré-venda cancelada e se imediatamente após a emissão deste Cupom Fiscal houve a emissão pelo ECF do Cupom Fiscal de Cancelamento.

Condição para requisito atendido: Emissão de Cupom Fiscal contendo os itens registrados na pré-venda cancelada e emissão do Cupom Fiscal de Cancelamento ou ainda o cancelamento de cupom em emissão, porém atendendo o item 3 do requisito V.

Condição para requisito não atendido: Efetivação do cancelamento da pré-venda sem a emissão do Cupom Fiscal e do Cupom Fiscal de Cancelamento ou a emissão do Cupom Fiscal sem a indicação do número do registro da pré-venda.

TESTE 006: Emissão de Redução Z com registro de pré-venda pendente

Passo 1: Com o ECF contendo Cupons Fiscais emitidos, registre no programa aplicativo uma operação de pré-venda com pelo menos um item e não emita o Cupom Fiscal respectivo deixando o registro de pré-venda pendente (dia D+0).

Passo 2: Emita a Redução Z (do dia D+0).

Passo 3: Avance a data do ECF para o dia seguinte ao da Redução Z emitida no passo 2 (dia D+1) e emita pelo menos um Cupom Fiscal, deixando o ECF com Redução Z do dia D+1 pendente.

Passo 4: Comande a emissão da Redução Z do dia D+1.

Passo 5: Observe se imediatamente antes da emissão da Redução Z do dia D+1, houve a emissão automática do Cupom Fiscal relativo ao registro de pré-venda pendente e em seguida o Cupom Fiscal Cancelamento.

Condição para requisito atendido: Emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente, imediatamente antes da emissão da Redução Z do dia D+1.

Condição para requisito não atendido: Falta de emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente, imediatamente antes da emissão da Redução Z do dia D+1.

TESTE 007: Abertura de movimento com registro de pré-venda pendente

Passo 1: Com o ECF contendo Cupons Fiscais emitidos, registre no programa aplicativo uma operação de pré-venda com pelo menos um item e não emita o Cupom Fiscal respectivo, deixando o registro de pré-venda pendente (dia D+0)

Passo 2: Emita a Redução Z (do dia D+0).

Passo 3: Avance a data do ECF para as 23h45min do dia seguinte ao da Redução Z emitida no passo 2 (dia D+1) e emita pelo menos um Cupom Fiscal, deixando o ECF com Redução Z do dia D+1 pendente.

Passo 4: Aguarde o ECF avançar até as 2h do dia seguinte ao da Redução Z pendente (dia D+2), verificando se a impressora entrou em estado de bloqueada (Redução Z pendente).

Passo 5: Observe se o ECF emitiu automaticamente a Redução Z pendente do dia D+1 e caso isto tenha ocorrido, verifique se, após a emissão da Redução Z, houve a emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente do movimento do dia D+0.

Passo 6: Caso o ECF não tenha emitido automaticamente a Redução Z pendente do dia D+1, comande a emissão desta Redução Z e verifique se, antes ou após a emissão da Redução Z, houve a emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente do movimento do dia D+0.

Condição para requisito atendido: Emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente.

Condição para requisito não atendido: Não emissão automática do Cupom Fiscal e seu cancelamento, relativos ao registro de pré-venda pendente.

TESTE 007A: Emissão de pré-venda sem registro contábil ou financeiro.

Passo 1: Emitir pelo Menu Fiscal o relatório MEIOS DE PAGAMENTO.

Passo 2: Registre uma pré-venda com 03 (três) itens, sendo o primeiro de R\$ 200,00(duzentos reais), o segundo de R\$ 30,00 (trinta reais) e o último de R\$ 150,00 (cento e cinquenta reais).

Passo 3: Emitir pelo Menu Fiscal o relatório MEIOS DE PAGAMENTO.

Condição para requisito atendido: Emissão do relatório MEIOS DE PAGAMENTO sem acréscimo dos valores definidos no Passo 2.

Condição para requisito não atendido: Emissão do relatório MEIOS DE PAGAMENTO com acréscimo de pelo menos um dos valores definidos no Passo 2.