FAPEG
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

PAPPE INTEGRAÇÃO

Manual do Processo de Teste de Sofware para Micro e Pequena Empresas versão 2.0

Goiânia
2013

PAPPE Integração

Agradecimentos

Esta conquista é resultado de apoio e ajuda de muitas pessoas que contribuíram para a realização deste trabalho.

À FAPEG e ao INF-UFG pelo auxílio financeiro e técnico.

Às empresas participantes e patrocinadoras deste projeto: Canion Software, Decisão Sistemas, Meta Tecnologia e Tron Informática.

Aos nossos parceiros: Mowe Tecnologia, TupiLabs e CERCOMP-UFG.

Aos colaboradores responsáveis pela execução deste projeto: Dr Auri Marcelo Rizzo Vincenzi, Dr Cássio Leonardo Rodrigues, Dr Celso Gonçalves Camilo Junior, Msc Jacson Rodrigues Barbosa, Adailton Ferreira de Araújo, Andressa Martins, Guilherme Sampaio Soares, Jailton Alkimin Louzada, Malba Jacob Prudente, Paulo Marcos Soares Rodrigues e Vinicius Vieira Pessoni.

Resumo

Integração, PAPPE. **Manual do Processo de Teste de Sofware para Micro e Pequena Empresas versão 2.0**. Goiânia, 2013. 57p. Manual Técnico. Instituto de Informática, Universidade Federal de Goiás.

Este manual objetiva apresentar uma proposta de Processo de Teste de Software no contexto de micro e pequenas empresas de TI (FreeTest).

Palavras–chave

Teste de software, Micro e Pequena Empresa de TI.

Conteúdo

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

Em uma década com emergente cultura web na qual lidar com imensos volumes de informação se tornou um paradigma natural, as indústrias de software experimentam diariamente um mercado incrivelmente dinâmico, vasto e extremamente competitivo. Nesse contexto, essas indústrias devem se esforçar em termos de especialização e atualização para sobressair frente à voraz concorrência. Além disso, para se manterem em suas posições de estabilidade devem oferecer aos seus clientes, os quais estão cada dia mais exigentes, produtos dotados do mais alto grau de qualidade.

A obtenção da alta qualidade dos produtos de software é influenciada por diversos fatores como escopo do produto, tempo disponível para execução do projeto, grau de conhecimento da equipe técnica, dificuldade do negócio, dentre vários outros. Dessa forma para que se obtenha produtos mais confiáveis, seguros, com melhor desempenho e que principalmente atendam bem às necessidades dos usuários é necessário que sua produção seja regida por métodos, padrões e também processos bem estruturados. Nesse sentido a área de Engenharia de Software tem profundas contribuições a fazer, bem como sua especialização e foco do projeto desenvolvido - o teste de software.

O processo de software define diretivas para que as equipes de desenvolvimento possam realizar com sucesso todas as etapas da produção de software, desde a concepção até o aceite do produto pelo cliente. Aliado a esse processo, preferencialmente em paralelo, deve existir um processo de teste, o qual possui a responsabilidade de auferir a qualidade tanto do processo quanto do produto em desenvolvimento.

Os benefícios esperados com a implantação de um processo de teste são inúmeros destacando dentre eles a obtenção de maior qualidade dos softwares produzidos pelas empresas além de redução do tempo gasto com manutenções corretivas e ainda a obtenção de métricas antes não coletadas. Informações como quantidade de defeitos por versão de produto, quanto do software foi testado, quantidade de ciclos entre o desenvolvimento e o teste auxiliam não só no melhor entendimento dos processos internos mas também aumentam a confiança da equipe no produto liberado para produção. Além dessas contribuições, o panorama que o teste expõe traz a luz questões outrora nem conhecidas e que são pertinentes para a tomada inteligente de decisões.

Em se tratando de software de qualidade, a realidade de nosso estado não fica atrás de outros polos do país, sendo anfitrião de empresas responsáveis por grandes soluções implantadas em diversos setores da economia. Nesse contexto, se faz extremamente necessário políticas adequadas para lidar com tal responsabilidade. Em especial na contribuição para maior qualidade final, quesito extremamente visado quando se trata de valor agregado de um produto, é que o teste de software desempenha papel fundamental.

Apesar de existirem inúmeras técnicas, políticas e metodologias na área de teste de software, no âmbito das micro e pequenas empresas existe uma deficiência perceptível em formas específicas para tais, em especial, no segmento de processos de teste de software e ferramentas que o apoiem de forma integrada. Tendo em vista esse panorama explanado é que surge o projeto "Estudo e Definição de Processo de Teste de Software para Micro e Pequenas Empresas de TI", por meio da iniciativa de profissionais do Instituto de Informática da Universidade Federal de Goiás com o apoio da FAPEG/FINEP criaram o programa PAPPE-Integração, em conjunto com grandes nomes da indústria de software do estado (Canion Software, Decisão Sistemas, Meta Tecnologia e Tron Informática). Essa união de esforços conflui então para a proposição de denominadores que atendam às necessidades identificadas, resultando assim em um processo de teste específico para micro e pequenas empresas adicionado de uma proposição de integração entre ferramentas amplamente utilizadas por especialistas da área.

1.1 Desafios do projeto

O processo de desenvolvimento de software está intimamente ligado a filosofia da empresa na qual está implantado e apesar da regra de ouro ser a a independência desse com as tecnologias vigentes ele é influenciado por essas tecnologias. Pontos como linguagens de programação, cultura, ferramentas, paradigmas, arquiteturas, metodologias exercem influência, em menor ou em maior grau, sobre o processo de software. Embora haja frameworks específicas e normas internacionais que regem a criação de processos, em cada empresa ele será dotado de peculiaridades e modificações que devem estar de acordo com a necessidade do negócio atendido por essas empresas.

Da mesma forma que o processo de desenvolvimento deve ser ajustado a realidade de onde é implantado o processo de teste também o é. Isso ocorre pois o teste possui ligação com o processo de desenvolvimento, e uma vez que o primeiro é fundamentalmente customizado o segundo também acompanha essa necessidade de ajustes próprios. Havendo observado esses pontos, vários desafios foram encontrados quanto ao desenvolvimento do projeto assim como em sua implantação nas empresas.

O primeiro dos desafios se encontra no campo do nível da similaridade entre as empresas. Apesar de todas se enquadrarem na categoria de micro ou pequenas empresas,

cada uma dessas possui finalidade de negócio, cultura e tecnologias distintas. Essas diferenças então dificultam a proposição de um processo único, que sirva sem modificações, para todas elas. Dado a heterogeneidade entre as empresas do consórcio, entende-se que cada empresa precisará, a partir do framework proposto de projeto, realizar ajustes para que o processo se encaixe melhor em suas realidades.

Outro desafio percebido, e bem conhecido da área de teste, se enquadra no campo das relações interpessoais entre equipe de desenvolvimento e equipe de teste. O foco é tratar da problemática encontrada quanto a forma de realizar a crítica ao trabalho dos profissionais envolvidos e os atritos que essa crítica pode gerar entre as equipes.

É importante entender que o teste de software não tem por objetivo primordial, como é de senso comum, mostrar que o produto está correto. Ao contrário do senso comum, o objetivo do teste é fazer o produto falhar, ou seja, descobrir, demonstrar e documentar os defeitos desse produto. Essa forma de enxergar o produto é conhecida como "natureza destrutiva do teste".

A problemática provinda dessa "natureza destrutiva" do teste é a colocação em evidência dos problemas, defeitos e inconformidades nos artefatos de colegas de equipe por meio de críticas, o que é no mínimo um assunto delicado. Esse apontamento exige bastante energia e sabedoria por parte de quem o realiza para que não seja confundido como um ataque pessoal ou desrespeito a qualidade do que foi produzido e competência de quem o fez. Caso essa crítica não seja realizada de forma correta, ela pode impactar de forma negativa, gerando atritos e espirito separatista em equipe, na produtividade ao invés de contribuir de forma positiva para essa.

Outro ponto que merece observação quanto a implantação dos testes é a impressão equivocada de que com a implantação dos testes todos os problemas e deficiências das empresas relacionados a produção de software serão resolvidos. Uma regra bem conhecida quando se trata da engenharia de software é que "Não existe bala de prata" (BROOKS JR., 1987), isso significa que não existe uma forma, uma metodologia, tecnologia ou regra mágica que seja adequada para todas as realidades e que resolva todos os problemas existentes no processo de software.

O teste de software pode ser enxergado como uma ferramenta em prol da qualidade o qual tem o papel de validar e verificar o que é produzido, influenciando positivamente na qualidade dos produtos. Porém, o teste sozinho não tem como resolver todos os problemas de qualidade, uma vez que essa está intimamente ligada a vários outros fatores. Assim, como qualquer outra ferramenta o teste deve ser utilizado em conjunto com outras políticas de forma correta e efetiva, para que sejam alcançados os resultados de melhoria esperados.

O último desafio, mas não menos importante, trata-se da mudança de cultura necessário para que o teste de software obtenha sucesso. Essa mudança de cultura pode ser

delineada no mais abrangente possível e deve incluir preferencialmente todas as camadas de produção de software. Nesse sentido, resistências foram percebidas quanto a mudança da forma de trabalho, novas práticas incluídas e documentações necessárias ao teste. Essa resistência é percebida principalmente devido a cultura imediatista que deve ser abrandada quando se fala em teste. O teste deve ser pensado como um investimento de médio a longo prazo, o qual irá com certeza produzir resultados positivos caso essa cultura seja corretamente implantada. Portanto, é necessário um esforço em conjunto, de todos os envolvidos com o processo de software, de modo que a cultura de teste seja incluída no dia a dia dos profissionais e esses a vivenciem de forma natural.

1.2 Objetivos do projeto

Criar e aplicar um processo de teste de software, adequado as empresas deste consórcio e similares, para melhorar a qualidade de seus produtos. Além disso, criar um software, integrador de ferramentas especializadas, que apoie o processo de teste.

1.3 Metodologia do projeto

A ênfase do projeto está em capacitar as micro e pequenas empresas em realizar testes de forma sistemática por meio da instanciação de um processo de teste com suporte automatizado. Nesse sentido, para o desenvolvimento do projeto, quatro macro atividades foram seguidas:

1. Treinamento: nessa atividade foram realizados o nivelamento e a capacitação das equipes envolvidas tanto nos conceitos teóricos de teste quanto práticos, incluindo a utilização de ferramentas de apoio e automatização dos teste. Além disso, foi previsto nessa atividade o treinamento da implantação do próprio processo de teste definido e das ferramentas de apoio integradas ao mesmo.
2. Processo de Teste: o objetivo foi a definição de um processo de teste genérico que possa, posteriormente, ser instanciado para cada empresa integrante do consórcio. Tal processo deve contemplar as atividades de projeto, execução e acompanhamento dos testes de unidade, integração, sistema e aceitação, e ser conduzido em paralelo ao processo de desenvolvimento de software. Essa colaboração entre o desenvolvimento e o teste minimiza o esforço e incentiva a detecção de defeitos nos diferentes artefatos nas fases iniciais do desenvolvimento, quando o custo de correção é menor. Em relação ao processo de teste de software (PT), alguns modelos de maturidade de PT foram propostos. Entre eles, TMM (Test Maturity Model), TMMi (Test Ma-

turity Model Integration) e MPT.Br (Melhoria do Processo de Teste de Software Brasileiro). Tais modelos foram considerados no desenvolvimento do projeto.

3. Integração/Desenvolvimento de ambiente de apoio: nesta atividade, de fundamental importância para o projeto, será definida uma arquitetura para a integração de ferramentas de teste de apoio às diferentes etapas do PT definido. Atualmente existe uma grande variedade de ferramentas de apoio a diferentes atividades pontuais do processo de teste, por exemplo, uma integração esperada é em relação a atividades de teste funcionais, gestão de testes e gestão de defeitos. Para cada uma destas atividades serão consideradas, preferencialmente, a adoção de ferramentas de código aberto para que os resultados fiquem à disposição não apenas das empresas proponentes, mas também de outras organizações interessadas em implantar e melhorar o seu PT. A integração das mesmas em um único ambiente é desejável para facilitar a adoção e reduzir a inconsistência de dados gerados, além de facilitar a rastreabilidade desses dados.
4. Suporte para a geração de documentos: documentar os dados produzidos durante os testes é de fundamental importância para se comprovar a execução das atividades exigidas pelo PT, principalmente na busca por certificações. Assim sendo, o objetivo é utilizar padrões previamente existentes, como a Norma IEEE 829-2008, para padronizar os documentos produzidos. Além disso, no desenvolvimento/integração das ferramentas mencionadas no Item 3, a ênfase será dada para automatizar a geração de tais documentos conforme o padrão escolhido, facilitando a rastreabilidade e consistência dos mesmos.

1.4 Organização do Manual

Este manual está organizado em 4 capítulos. Nesse capítulo foram apresentados os principais conceitos relacionados a essa pesquisa, o objetivo e a metodologia do projeto visando delinear o tema explorado neste manual. No entanto, o escopo deste manual é apenas a apresentação da proposta de Processo de Teste de Software para Micro e Pequenas Empresas (FreeTest).

O Capítulo 2 apresenta os conceitos gerais sobre o FreeTest.

No Capítulo 3 são apresentadas as práticas específicas de cada uma das áreas de processo do FreeTest.

Já no Capítulo 4 são feitas considerações finais, apresentadas as limitações e os possíveis desdobramentos decorrente do projeto.

No Apêndice A é apresentado o Glossário com as respectivas definições dos principais termos utilizados no trabalho.

O Apêndice B apresenta uma proposta de *template* para elaboração do Plano de Teste.

Já o Apêndice C propõe um *template* para definição dos Cenários de Teste.

Finalmente, no Apêndice D é apresentado um *template* para apoiar a confecção do Relatório de Consolidação de Teste.

CAPÍTULO 2

Visão Geral do FreeTest

Modelos de melhoria de processos de software, tais como o CMM (*Capability Maturity Model*) e o CMMI (*Capability Maturity Model Integration*), não têm dado uma importância significativa aos processos de V&V (Validação e Verificação). Em decorrência disso os conceitos relacionados aos processos de maturidade de teste não são bem definidos (BURNSTEIN et al., 1996), consequentemente as organizações produtoras de software acabam executando estes processos de forma inadequada (JACOBS; TRIENEKENS, 2002), ou mesmo ficando desorientadas em como implantar um processo de teste de software adequadamente.

Como uma tentativa de demandar mais atenção aos processos de V&V, foi proposto o TMM (*Testing Maturity Model*), que foi desenvolvido por um grupo de pesquisa do *Illinois Institute of Tecnology* (BURNSTEIN, 2003).

O TMM tem como objetivo ser um complemento aos modelos de capacidade e maturidade, mais especificamente visa disponibilizar processos de V&V bem definidos, para que os gerentes de teste e os grupos de garantia da qualidade das organizações possam tê-lo como orientação (BURNSTEIN et al., 1996).

Não apenas o acrônimo TMM se assemelha ao CMM, mas também os conceitos essenciais existentes em ambos são bastante próximos. Por exemplo, no TMM faz-se uso do conceito de nível de maturidade como roteiro para melhorar e avaliar os processos de teste, possuindo assim também, 5 níveis de maturidade.

Também tentando complementar os modelos de melhoria de processo, recentemente foi proposto o TMMi (*Test Maturity Model integration*) que utiliza o TMM como uma das principais fontes. Além do TMM, foi largamente guiado pelo trabalho feito no CMMI. O TMM e o TMMi foram definidos como modelos a serem implementados por estágios, sendo que o TMMi define um conjunto de áreas de processos que orientam o caminho de melhoria da organização (VEENENDAAL, 2012).

Como uma iniciativa brasileira, foi proposto o modelo Melhoria de Processo de Teste Brasileiro (MPT.Br) que trata a melhoria do processo de teste por meio de práticas relativas às atividades desenvolvidas ao longo do ciclo de vida de teste do produto (SOFTEXRECIFE; RIOSOFT, 2011). É importante ressaltar que o MPT.Br tomou como base

diversos modelos de referência em teste de software e modelos de referência em melhoria de processo de software, tais como: TMM, TMMi, CMMI, MPS.BR e outros.

Na Tabela 2.1 é apresentado o mapeamento entre o TMMi e o MPT.Br, demonstrando assim a equivalência entre os mesmos.

Dentre os modelos de maturidade de processo de software tais como TMMi e MPT.Br, pouco se sabe sobre suas implementações no contexto de empresas de micro e pequeno porte como as que compõem o projeto proposto. Por isso, o projeto Estudo e Definição de Processo de Teste de Software para Micro e Pequenas Empresas de TI visa agrupar diferentes atividades de verificação e validação em um Processo de Teste de Software para Micro e Pequenas Empresas (FreeTest) coerente para o contexto.

2.1 Organização do FreeTest

O FreeTest foi definido a partir dos modelos de maturidade TMMi e MPT.Br, sendo portanto organizado em níveis de maturidade. Cada nível de maturidade é composto por um conjunto de áreas de processo. Uma área de processo corresponde a um conjunto de práticas específicas que individualmente objetivam atender metas específicas.

Os níveis de maturidade definem patamares de evolução de processos, caracterizando estágios de melhoria da implementação de processos na organização. O nível de maturidade em que se encontra uma organização permite prever o seu desempenho futuro ao executar um ou mais processos (ROCHA; MACHADO, 2011). A escala de maturidade do FreeTest se inicia no nível E e evolui até o nível A, na Tabela 2.2 é apresentada a organização em níveis e as correspondentes áreas de processo. Já na Tabela 2.3 é apresentado o mapeamento entre o FreeTest e o MPT.Br.

2.2 Fluxo Principal do FreeTest

A Figura 2.1 apresenta o fluxo principal do FreeTest. Quando uma solicitação (demanda) de modificação é cadastrada por meio do processo de suporte e a mesma é emergencial, inicia-se imediamente a referida alteração do produto de software por meio do processo de desenvolvimento. Depois de concluído a realização dos testes unitários é iniciado a execução dos testes funcionais, caso seja identificado algum defeito durante a execução dos testes, o mesmo é registrado e encaminhado para o processo de desenvolvimento. Após finalizar a realização dos testes funcionais o produto de software com as devidas alterações é encaminhado para a gerência de configuração e mudanças (GCM).

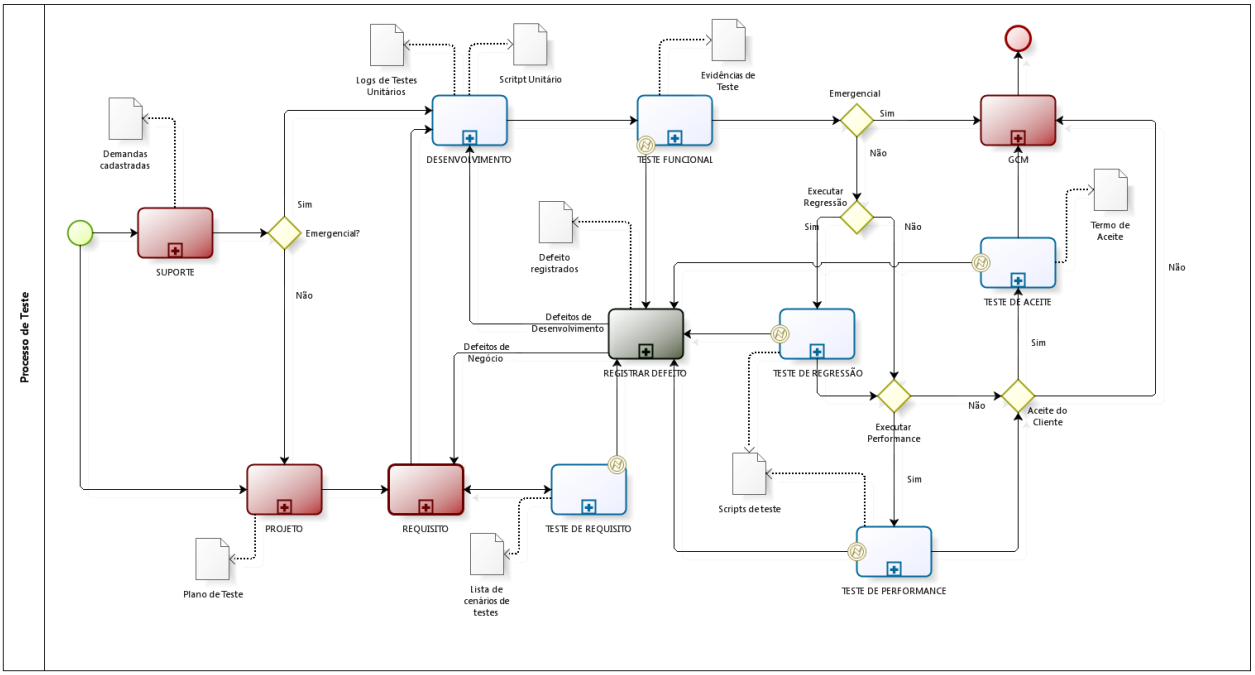

Figura 2.1: *Fluxo principal do FreeTest*

Tabela 2.1: *Mapeamento entre as áreas de processo do modelo TMMi e do MPT.Br*

Modelo TMMi		**Modelo MPT.Br**	
Nível	**Áreas de Processo**	**Nível**	**Áreas de Processo**
2	Estratégias e políticas de teste	1	Gerência de Projetos de Teste (prática específica GPT1)
2	Planejamento de teste	1	Gerência de Projetos de Teste (práticas específicas GPT2 a GPT20)
		2	Gerência de Projetos de Teste (práticas específicas GPT21 a GPT25)
		2	Gerência de Requisitos de Teste (práticas específicas GRT1 a GRT5)
2	Execução e projeto de teste	1	Projeto e execução de Teste (práticas específicas PET1 a PET4)
		2	Projeto e Execução de Teste (práticas específicas PET5 e PET6)
		3	Projeto e Execução de Teste (prática específica PET7)
2	Ambiente de Teste	3	Gerência de Projetos de Teste (práticas específicas GPT26 a GPT28)
2	Controle e monitoramento do Teste	3	Fechamento do Teste (práticas específicas FDT1 a FDT4)
		3	Garantia da Qualidade (práticas específicas GDQ1 a GDQ3)
3	Teste não-funcional	4	Teste Não-Funcional (práticas específicas TNF1 a TNF3)
3	Organização de teste	3	Organização do Teste (práticas específicas OGT1 a OGT10)
3	Ciclo de vida de teste e integração		
3	Programa de treinamento de teste	3	Treinamento (práticas específicas TRE1 a TRE4)
3	Revisão em pares	3	Teste Estático (práticas específicas TES1 a TES5)
4	Revisões Avançadas	3	Teste de Aceitação (práticas específicas TDA1 a TDA7)
4	Medição de teste	3	Medição e Análise de Teste (práticas específicas MAT1 a MAT5)
4	Avaliação da qualidade de produto	4	Avaliação da Qualidade do Produto (práticas específicas AQP1 a AQP5)
5	Prevenção de defeitos	4	Gestão de Defeitos (práticas específicas GDD1 a GDD3)
5	Controle de qualidade	5	Controle Estatístico do Processo (práticas específicas CEP1 a CEP5)
5	Otimização do processo de teste	5	Automação da Execução do Teste (práticas específicas AET1 a AET6)
		5	Gestão de Ferramentas (práticas específicas GDF1 a GDF6)
		4	Organização do Teste (práticas específicas OGT11 e OGT12)

Tabela 2.2: *Níveis do FreeTest*

Nível de Maturidade	Áreas de Processo	Práticas Específicas
A	TPE - Teste de desempenho	TPE1 - Preparar massa de teste
		TPE2 – Manter script de desempenho
		TPE3 – Executar teste de desempenho
		TPE4 – Encerrar teste de desempenho
B	INC – Integração contínua	INC1 – Elaborar código
		INC2 – Elaborar scripts BDD
		INC3 – Montar build
C	TRG – Teste de regressão	TRG1 – Preparar massa de teste
		TRG2 – Manter script de regressão
		TRG3 – Executar teste de regressão
		TRG4 – Encerrar teste de regressão
D	TDA – Teste de aceite	TDA1 – Atualizar ambiente de teste
		TDA2 – Executar teste de aceite
		TDA3 – Encerrar teste de aceite
	TRQ – Teste de requisito	TRQ1 – Realizar verificação
		TRQ2 – Encerrar verificação
E	TFU – Teste funcional	TFU1 – Atualizar ambiente de teste
		TFU2 – Realizar teste
		TFU3 – Encerrar teste
	GPT – Gerência de Projetos de teste	GPT1 – Elaborar plano de teste

Tabela 2.3: *Mapeamento entre o Modelo FreeTest e o MPT.Br*

Modelo FreeTest		Modelo MPT.Br	
Nível	**Áreas de Processo**	**Nível**	**Áreas de Processo**
A	Teste de desempenho	4	Avaliação da Qualidade do Produto (práticas específicas AQP1 a AQP5)
B	Integração contínua	5	Automação da Execução do Teste (práticas específicas AET1 a AET6)
C	Teste de regressão	3	Organização do Teste (prática específica OGT10)
D	Teste de aceite	3	Teste de Aceitação (práticas específicas TDA1 a TDA7)
	Teste de requisito	3	Projeto e Execução de Teste (prática específica PET7)
E	Gerência de Projetos de teste	1	Gerência de Projetos de Teste (práticas específicas GPT1 a GPT20)
	Teste funcional	1	Projeto e execução de Teste (práticas específicas PET1 a PET4)

CAPÍTULO 3

Áreas de processo do FreeTest

3.1 Gerência de projetos de teste (GPT)

A área de processo de gerência de projetos de teste possui apenas uma prática específica (elaborar plano de teste (GPT1)). O principal objetivo dessa área de processo é apoiar a integração dos planos, permitindo assim a sincronização entre o plano de teste e os demais (plano de projeto, plano de gerência de requisitos e plano de gerência de configuração).

O Apêndice B apresenta uma sugestão para confecção do Plano de Teste.

3.1.1 Elaborar plano de teste (GPT1)

A Tabela 3.1 apresenta uma descrição da prática elaborar plano de teste. O objetivo dessa prática é planejar os testes a serem realizados ao longo do processo de desenvolvimento.

Figura 3.1: *Fluxograma da prática elaborar plano de teste*

3.2 Teste funcional (TFU)

Conforme é apresentado na Figura 3.2, a área de processo de teste funcional tem como objetivo viabilizar a preparação do ambiente de teste (atualizar ambiente de teste (TFU1)) para a efetiva execução dos mesmos (realizar teste (TFU2)). Após os testes terem sido executados de forma satisfatória, ocorre a finalização de sua execução (encerrar teste (TFU3)).

Tabela 3.1: *Descrição da prática elaborar plano de teste*

<table>
<tr><td>Objetivos</td><td colspan="3">Realizar o planejamento dos testes a serem executados sobre as demandas elencadas para atendimento da versão.</td></tr>
<tr><td>Participantes</td><td>Responsável:
Gerente de Teste</td><td>Colaborador:
Gerente de Projetos</td><td>Aprovação:</td></tr>
<tr><td rowspan="2">Entradas</td><td colspan="3">Critérios:
- Projeto da versão definido;
- Demanda de Planejamento de Teste com status “Nova” ou “Aguardando”.</td></tr>
<tr><td colspan="3">Artefatos:
- Documento de Projeto;
- DA.0001 – Padrões de Nomenclatura.</td></tr>
<tr><td>Ações</td><td colspan="3">1-Havendo alguma inconformidade na abertura da demanda:
1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes.
2-Alterar o status da demanda de Planejamento de Teste para “Em andamento”;
3-Recuperar o Documento de Projeto no devido repositório da versão;
4-Recuperar o modelo do RQ.0001 no devido repositório de templates;
5-Elaborar o RQ.0001 conforme informações contidas no Documento de Projeto;
6-Armazenar o RQ.0001 no devido diretório de testes existente no repositório da versão, fazendo uso dos padrões de nomenclatura citados no DA.0001;
7-Encerrar a Demanda de Planejamento de Teste lançando o devido esforço utilizado na execução da tarefa.</td></tr>
<tr><td rowspan="2">Saídas</td><td colspan="3">Critérios:
- Planejamento dos testes definido;
- Demanda de Planejamento de Testes com status “Resolvido”.</td></tr>
<tr><td colspan="3">Produtos:
- RQ.0001 – Plano de Teste.</td></tr>
</table>

3.2.1 Atualizar ambiente de teste (TFU1)

A atualização do ambiente de teste visa que a equipe de testes tenham em seu ambiente a disponibilização da última versão do software, que será submetida a uma sequência de testes funcionais. Este ambiente deve ser controlado de forma a não sobrepor versões quando outra estiver com atividades de testes em execução, evitando que seja relatado defeitos indevidos (oriundos de inconformidade com requisitos ou integração com outros módulos, etc).

3.2.2 Realizar teste (TFU2)

A realização dos testes deverá ocorrer assim que uma versão do software for disponibilizada no ambiente de teste. Os testes a serem realizados devem ser indicados no documento “Notas da Versão”, focando inicialmente no funcionamento básico das funcionalidades e posteriormente havendo prazo hábil a execução de testes complementares, cenários estes mapeados no documento de Cenários de Testes. Havendo defeitos identificados, esses devem ser registrados e acompanhados até a entrega com as devidas correções. A Tabela 3.2 descreve detalhes sobre a prática realizar teste.

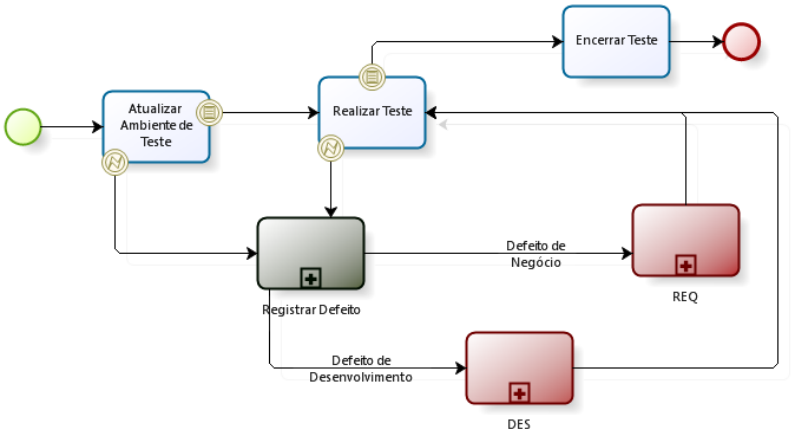

Figura 3.2: *Fluxograma da área de processo teste funcional*

3.2.3 Encerrar teste (TFU3)

Após a conclusão dos testes, caberá ao Gerente de Teste realizar a consolidação dos resultados da execução dos testes. Diante disso ter indicadores suficientes para discutir com os *stakeholders* envolvidos sobre a qualidade do produto e consequente disponibilização em ambiente de produção. A Tabela 3.3 descreve detalhes sobre a prática encerrar teste.

3.3 Teste de requisito (TRQ)

A área de processo de teste de requisitos (apresentada na Figura 3.3) visa identificar possíveis inconsistências existentes nos requisitos do software (realizar verificação (TRQ1)) por meio da confecção/revisão das descrições dos cenários de teste.

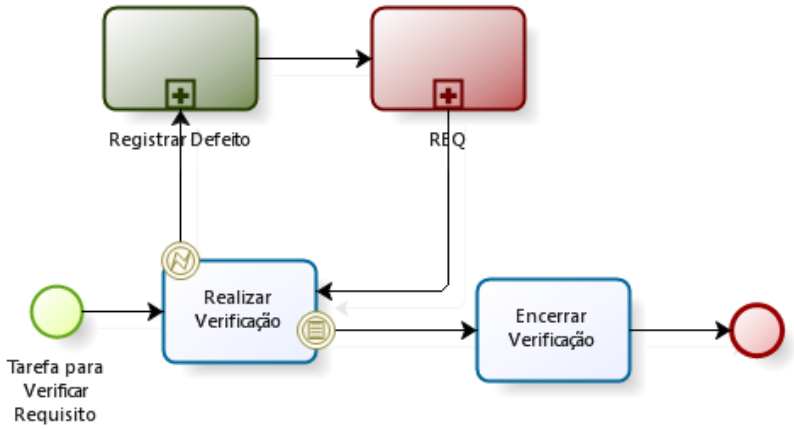

Figura 3.3: *Fluxograma da área de processo teste de requisito*

3.3.1 Realizar verificação (TRQ1)

Para desenvolver os casos de teste, os responsáveis por esse necessitam obter os artefatos relativos a descrição do software - documento de requisitos, casos de uso, diagrama de sequência, protótipos de tela, dentre vários outros. Dessa forma, enquanto é realizado o desenvolvimento dos casos de teste uma revisão dos artefatos obtidos é também realizada. Ainda que seja somente por meio da leitura para obter informações do produto, essa revisão pode contribuir para a qualidade desses artefatos reportando qualquer defeito/inconsistência encontrada.

A Tabela 3.4 descreve detalhes sobre a prática realizar verificação.

3.3.2 Encerrar verificação (TRQ2)

Após a conclusão das verificações, caberá ao Gerente de Teste realizar a consolidação dos resultados da verificação para obter informações sobre a qualidade da documentação produzida durante a fase de análise do software. A Tabela 3.5 apresenta detalhes sobre a prática encerrar verificação.

3.4 Teste de aceite (TDA)

A Figura 3.4 apresenta a visão geral da área de processo teste de aceite, cujo o principal objetivo é validar uma *baseline* do produto de software com o cliente.

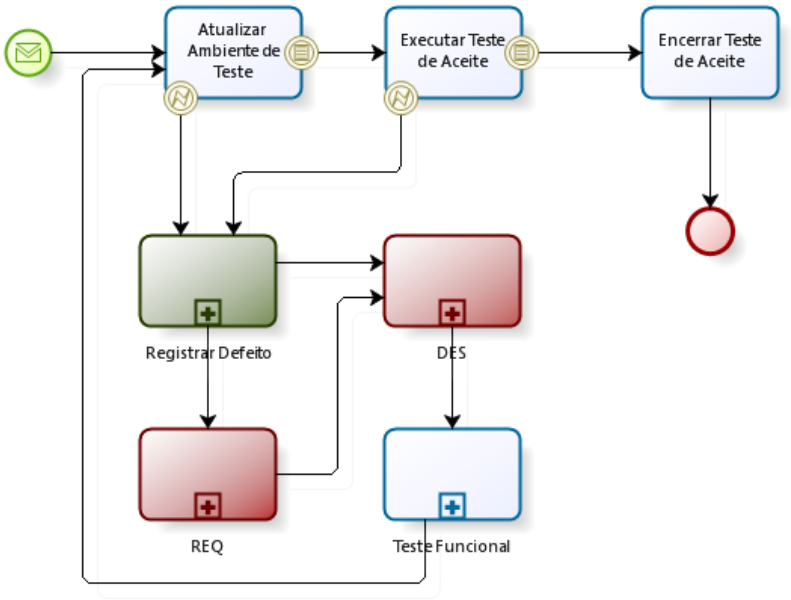

Figura 3.4: *Fluxograma da área de processo teste de aceite*

3.4.1 Atualizar ambiente de teste (TDA1)

A atualização do ambiente de teste para realização dos testes de aceite, objetiva viabilizar que os stakeholders envolvidos com o teste tenham acesso à última versão do software. Sempre que possível o teste de Aceite deve ocorrer em ambiente totalmente independente visando evitar concorrência com outras atividades de desenvolvimento ou testes, não sendo possível é importante planejar e controlar todas atividades a serem executadas por meio do gerenciamento de risco. A Tabela 3.6 apresenta detalhes sobre a prática atualizar ambiente de teste.

3.4.2 Executar teste de aceite (TDA2)

Dependendo da forma como o ambiente de teste de aceite foi construído, os stakeholders poderão realizar os testes presencialmente ou remotamente. Esse testes objetivam garantir que o software atenda as necessidades fundamentais do negócio, não se preocupando com aspectos irrelevantes ao negócio. Também é importante que esses testes sejam acompanhados por profissionais de outras áreas da Tecnologia da Informação (por exemplo, Negócio, Análise e Desenvolvimento), afim de dar todo suporte necessário aos stakeholders. A Tabela 3.7 apresenta detalhes sobre a prática executar teste de teste.

3.4.3 Encerrar teste de aceite (TDA3)

Os testes de aceite serão considerados concluídos quando as metas de qualidade (definidas no plano do projeto) forem atendidas. Devendo também ser produzido e assinado pelos stakeholders um termo de aceite do produto validado bem como um relatório consolidado indicando todas as inconformidades identificadas, corrigidas e melhorias futuras, indicadores esses que deverão ser discutidos posteriormente com os stakeholder envolvidos. A Tabela 3.8 apresenta detalhes sobre a prática encerrar teste de teste.

3.5 Teste de regressão (TRG)

A área de processo de teste de regressão (apresentada na Figura 3.5) tem como meta permitir a realização de testes para garantir que partes do software que foram alteradas não introduziram novos defeitos no mesmo.

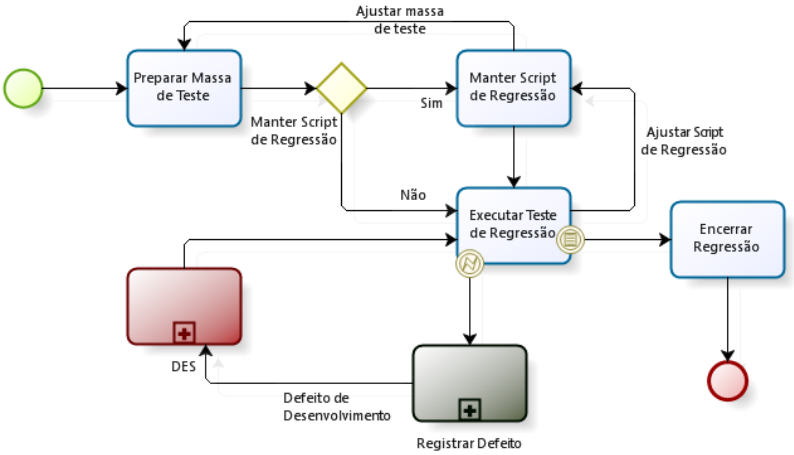

Figura 3.5: *Fluxograma da área de processo teste de regressão*

3.5.1 Preparar massa de teste (TRG1)

A atividade de preparação da massa de teste consiste na criação ou ajuste dos dados que serão utilizados para a execução dos testes automatizados. Uma adequação nos dados deve ser realizada pelo profissional responsável para que seja garantido a máxima eficácia desses dados. Por eficácia, entende-se uma maior probabilidade de provocar a revelação de defeitos e a detecção de falhas. Para fazer essa adequação, pode-se utilizar técnicas de derivação de casos de teste, tais como: análise de valor limite e partição de equivalência.

Caso esses dados estejam em uma base de dados, um conhecimento prévio da linguagem SQL é necessário, para extrair ou inserir dados na respectiva base. É importante ressaltar que essas informações devem estar com tipos de dados compatíveis aos esperados pelo sistema, por exemplo, para um atributo data, deve-se ter massa de dados compatível com o tipo data. Na Tabela 3.9 é apresentado mais detalhes sobre a respectiva prática.

3.5.2 Manter script de regressão (TRG2)

A repetição da execução dos testes de maneira exaustiva, ou seja de todo o sistema, é extremamente onerosa. Isso ocorre diversas vezes por não se saber ao certo os impactos causados por determiandas mudanças, ou mesmo por haver algumas áreas não cobertas no produto sob teste. Como uma forma de diminuição desse custo de execução procura-se, sempre que possível, a utillização de testes automatizados.

Para a execução dessas ferramentas, são necessários scripts. Esses scripts são programas simples escritos em linguagens comerciais, que uma linguagem estruturada, que permite executar as ações previamente capturadas pelo usuário. Cada ferramenta tem sua própria linguagem e formas de capturar os elementos que interagem com o usuário,

mas em linhas gerais, sempre que algum elemento alvo da automação é modificado é necessário alterar também o script de teste. Essas manutenções nos scripts garantem a integridade dos mesmos e uma execução sem interrupções.

As manutenções nos scripts de teste devem ser atividades corriqueiras na organização. Uma vez que o acúmulo dessa atividade pode dificultar a manutenção e/ou comprometer partes íntegras dos mesmos. Pede-se que sempre antes de realizar um teste de regressão verificar os scripts e massas utilizadas, para que não ocorra quebra no fluxo normal da execução automatizada. Na Tabela 3.10 é apresentado mais detalhes sobre a respectiva prática.

3.5.3 Executar teste de regressão (TRG3)

Segundo (MULLER, 2011), "Teste de regressão é o teste repetido de um programa que já foi testado, após sua modificação, para descobrir a existência de algum defeito introduzido ou não coberto originalmente como resultado da mudança".

O teste de regressão, por ser um teste caro e demorado, normalmente, é executado por ferramentas de automação de teste. Este tipo de teste consiste em re-executar uma parte significativa ou todo o sistema de modo a verificar se não houve introdução de defeitos após correções no software. Em alguns casos, dependendo da tecnologia utilizada no desenvolvimento, uma única vírgula mal colocada ou faltante pode provocar falhas na execução do software e/ou acarretar comportamentos inadequados em partes já testadas do programa por exemplo. Para garantir que a correção de uma falha não irá provocar um comportamento inesperado, frequentemente é realizado o reteste de todo o sistema ou as partes mais propensas a erros.

O FreeTest defende a visão de que falhas encontradas em produção devem ser automatizadas e armazenadas em um repositório para posterior execução, criando assim, uma lista de casos de testes para regressão. Esta lista para regressão tem o propósito de assegurar que os incidentes encontrados pelos usuários finais não se repetirão, garantindo a confiabilidade do produto pelo cliente. Na Tabela 3.11 é apresentado mais detalhes sobre a respectiva prática.

3.5.4 Encerrar teste de regressão (TRG4)

O FreeTest defende a ideia que, ao finalizar os testes de regressão deve-se apresentar um relatório da execução dos testes. Esse relatório é uma forma de avaliar a qualidade e criar confiabilidade no produto que está sendo produzido. Mesmo que haja defeitos ainda a serem consertados, o responsável pelo produto tem a exata noção de como anda a qualidade, podendo tomar as devidas ações em casos de desvios da qualidade ou, em casos extremos, abortar a produção do sistema.

Normalmente as ferramentas de automação utilizadas nos testes de regressão disponibilizam relatórios para as análises dos resultados, no entanto é responsabilidade da organização definir quais as métricas e indicadores serão mais adequados às suas necessidades. Na Tabela 3.12 é apresentado mais detalhes sobre a respectiva prática.

3.6 Integração contínua (INC)

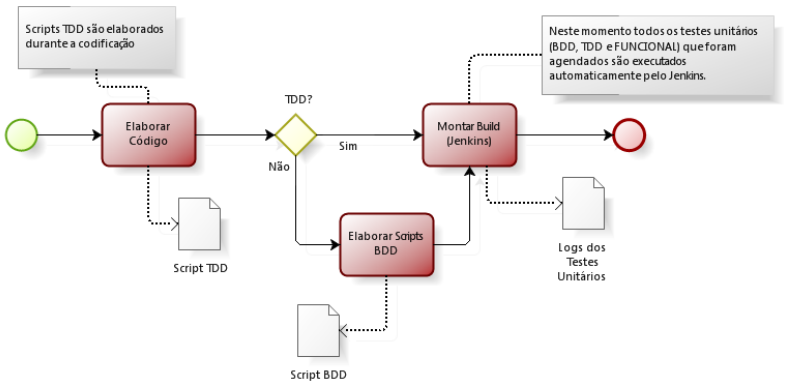

Figura 3.6: *Fluxograma da área de processo integração contínua*

3.6.1 Elaborar código (INC1)

3.6.2 Elaborar script BDD (INC2)

3.6.3 Montar build (INC3)

3.7 Teste de desempenho (TPE)

A área de processo de teste de desempenho tem como alvo identificar métricas importantes ao software em questão visando monitorar por meio dessas os atributos de qualidade do software. A Figura 3.7 apresenta o relacionamento entre as práticas específicas da área de processo de teste de desempenho.

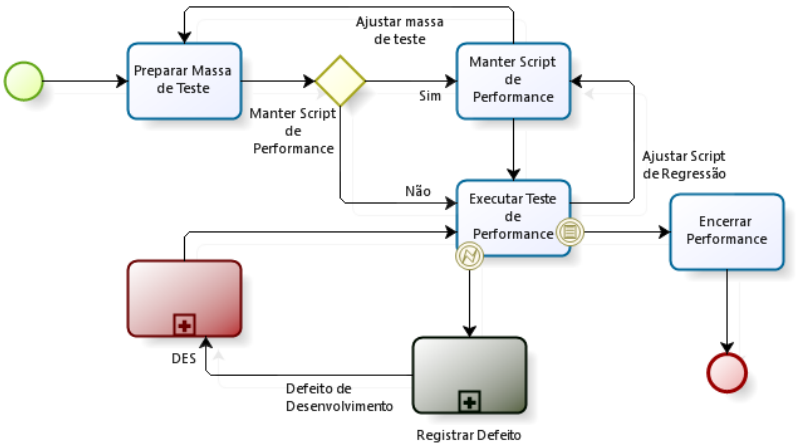

Figura 3.7: *Fluxograma da área de processo teste de desempenho*

3.7.1 Preparar massa de teste (TPE1)

Após a conclusão do planejamento dos testes de desempenho e a identificação dos pontos chaves do software que requeiram a análise de desempenho, será preciso elaborar uma massa de testes que viabilize sua execução, sendo esta adequada para medir o desempenho esperado da aplicação, conforme indicador estabelecido no início do projeto. Sempre que necessário esta massa deverá sofrer ajustes e manutenções necessárias.

3.7.2 Manter script de desempenho (TPE2)

Concluída a preparação das massas de testes, deverá ser providenciado a confecção dos scritps que irão realizar a avaliação de desempenho da aplicação. Estes scripts deverão estar armazenados em um repositório específico, seguindo também uma nomenclatura padronizada. Na Tabela 3.13 é apresentado mais detalhes sobre a respectiva prática.

3.7.3 Executar teste de desempenho (TPE3)

A execução dos testes de desempenho devem ocorrer a medida que houver alguma mudança nos pontos identificados na aplicação, bem como quando da ocorrência de mudanças estruturais do software (banco de dados, sistema operacional, servidores, etc.). Conforme resultados obtidos ou defeitos identificados, os ajustes deverão ser providenciados até que os indicadores esperados sejam atingidos. Na Tabela 3.14 é apresentado mais detalhes sobre a respectiva prática.

3.7.4 Encerrar teste de desempenho (TPE4)

Estando os testes de desempenho concluídos, seus resultados deverão ser compilados de forma a identificar os defeitos que eventualmente tenha ocorridos, bem como o plano de ação aplicado para resolução destes. Com isso haverá condições de se obter o nível de qualidade sobre o aspecto de desempenho da software. Na Tabela 3.15 é apresentado mais detalhes sobre a respectiva prática.

Tabela 3.2: *Descrição da prática realizar teste*

Objetivos	Realizar os testes funcionais de forma manual para a nova versão da aplicação.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Ambiente de teste disponível com nova versão da aplicação; - Demanda de Execução de Teste com status “Em andamento”.		
	Insumos: - Documentação produzida pela Análise; - DA.0001 – Padrões de Nomenclatura; - DA.0002 – Padrão para Registro de Defeito.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar os RQs (Requisitos) disponibilizados no devido repositório do release; 3-Recuperar o RQ.0002 contendo os cenários de testes previamente identificados no devido repositório do release; 4-Havendo necessidade, realizar os ajustes nos cenários de testes e atualizá-los no devido repositório do release; 4.1-Caso tenha necessidade de testes de desempenho, deverá ser elaborado o cenário específico para execução destes testes. 5-Executar os testes funcionais sobre a aplicação reportando os resultados esperados na ferramenta de gestão de testes; 6-Havendo algum defeito a ser registrado: 6.1-Abrir uma demanda de defeito conforme instruções contidas no DA.0002; 7-Resolver a demanda de Execução de Teste lançando o devido esforço destinado à execução da tarefa. 8-Caso a demanda esteja marcada para realização dos testes de regressão: 8.1-Abrir uma demanda de Teste de Regressão, vinculada à demanda de Execução de Teste e atribuir ao responsável. 9-Caso a demanda esteja marcada para realização dos testes de desempenho: 9.1-Abrir uma demanda de Teste de desempenho, vinculada à demanda de Execução de Teste e atribuir ao responsável.		
Saídas	**Critérios:** - Execução de testes concluída e resultados devidamente lançados na ferramenta de gestão de testes; - Não haver defeitos críticos em aberto; - Demanda de execução de teste com status “Resolvido”; - Demanda de Teste de Regressão com status “Aberto” (se houver necessidade); - Demanda de Teste de desempenho com status “Aberto” (se houver necessidade).		
	Produtos: - RQ.0002 – Cenários de Testes; - Registros de defeitos.		

Tabela 3.3: *Descrição da prática encerrar teste*

Objetivos	Elaborar um relatório conclusivo sobre a execução dos testes manuais do release disponibilizado para testes, a fim de obter um laudo da qualidade da aplicação.		
Participantes	**Responsável:** Gerente de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Execução dos testes manuais concluída. - Demanda de Execução de Teste com status “Resolvido”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - Registros de defeitos.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0003 no devido repositório de templates; 3-Validar e consolidar os resultados obtidos no RQ.0003 na área destinada a Teste de Manuais; 3.1-Caso necessário, atribuir a demanda de volta ao Analista de Teste responsável para devidos ajustes. 4-Armazenar o RQ.0003 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Fechar as demandas de Execução de Testes lançando o devido esforço utilizado na execução da tarefa.		
Saídas	**Critérios:** - Consolidação de resultados da Execução de Testes concluída. - Demandas de Execução de Teste com status “Fechado”.		
	Produtos: - RQ.0003 – Relatório Consolidado de Teste.		

Tabela 3.4: *Descrição da prática realizar verificação*

Objetivos	Realizar a verificação dos requisitos a fim de identificar possíveis inconsistências de negócio que podem levar a falhas futuramente.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Documentos de Requisitos concluídos; - Demanda de Verificação de Requisito com status “Nova” ou “Aguardando”.		
	Insumos: - Documentação produzida pela Análise; - DA.0001 – Padrões de Nomenclatura; - DA.0002 – Padrões para Registro de Defeito.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Alterar o status da demanda de verificação de requisito para “Em andamento”; 3-Recuperar os RQs (Requisitos) disponibilizados no devido repositório do release; 4-Recuperar o modelo do RQ.0002 no devido repositório de templates; 5-Avaliar cada requisito disponibilizado para verificação; 6-Havendo algum defeito a ser registrado: 6.1-Abrir uma demanda de defeito conforme instruções contidas no DA.0002; 7-Elaborar RQ.0002 contendo os possíveis cenários de testes a serem realizados; 8-Armazenar o RQ.0002 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 9-Resolver a demanda de Verificação de Requisitos lançando o devido esforço destinado à execução da tarefa.		
Saídas	**Critérios:** - Verificação de requisitos concluída; - Não haver defeitos críticos em aberto; - Tarefa de Verificação de Requisitos com status igual a “Resolvido”.		
	Produtos: - RQ.0002 – Cenários de Testes; - Registros de defeitos.		

Tabela 3.5: *Descrição da prática encerrar verificação*

Objetivos	Elaborar um relatório conclusivo sobre a execução das verificações dos requisitos disponibilizados, a fim de obter um laudo da qualidade dos requisitos.		
Participantes	**Responsável:** Gerente de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Verificação de Requisitos concluída; - Tarefas de Verificação de Requisito com status igual a “Resolvido”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - Registros de defeitos.		
Ações	1-Recuperar o modelo de RQ.0003 no devido repositório de templates; 2-Validar e consolidar os resultados obtidos no RQ.0003; 3-Armazenar o RQ.0003 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 4-Fechar as demandas de Verificação de Requisito lançando o devido esforço utilizado na execução da tarefa.		
Saídas	**Critérios:** - Consolidação de resultados da Verificação de Requisitos concluída; - Demandas de Verificação de Requisito com status “Fechado”.		
	Produtos: - RQ.0003 – Relatório Consolidado de Teste.		

Tabela 3.6: *Descrição da prática atualizar ambiente de teste*

Objetivos	Realizar a atualização controlada do ambiente de teste/aceite com a última versão do software liberado para execução das atividades de testes funcionais e de aceite.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Tarefas de desenvolvimento devidamente concluídas; - Permissão de atualização por parte do gerente/líder de teste; - Demanda de Execução de Teste com status “Nova” ou “Aguardando”.		
	Insumos: - DA.0002 – Padrão para Registro de Defeito; - Nota de Release; - Pacote da nova versão do software.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 2-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 3-Alterar o status da demanda execução de teste para “Em andamento”; 4-Caso tenha necessidade de atualização do banco de dados, executar os procedimentos pertinentes para sua atualização ou encaminhar para a área responsável; 5-Recuperar os pacotes da nova versão do software disponibilizado no devido repositório do release, e realizar os procedimentos necessários para atualização da aplicação, respeitando a tabela de horários determinada para execução esta atividade; 6-Realizar a validação do ambiente de teste, certificando que o mesmo está funcional; 7-Havendo algum defeito a ser registrado: 7.1-Abrir uma demanda de defeito na conforme instruções contidas no DA.0002. 8-Na demanda de Execução de Teste lançar o devido esforço utilizado na execução da atividade.		
Saídas	**Critérios:** - Aplicação funcionando adequadamente em ambiente de teste; - Demanda de Execução de Teste com status “Em andamento”; - Não haver defeitos críticos em aberto.		
	Produtos: - Registros de defeitos; - Ambiente de teste devidamente atualizado.		

Tabela 3.7: *Descrição da prática executar teste de aceite*

Objetivos	Envolver os clientes na homologação do produto que será entregue em ambiente de produção, garantindo que esse atenderá suas necessidade e expectativas.		
Participantes	**Responsável:** Cliente	**Colaborador:** Analista de Teste	**Aprovação:**
Entradas	**Critérios:** - Ambiente de teste disponível com nova versão da aplicação; - Demanda de Execução de Aceite com status “Em andamento”.		
	Insumos: - Documentação produzida pela Análise; - DA.0001 – Padrões de Nomenclatura; - DA.0002 – Padrões para Registro de Defeito.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar os RQs (Requisitos) disponibilizados no devido repositório do release; 3-Acompanhar e auxiliar na execução dos testes de aceite sobre a aplicação; 4-Havendo algum defeito a ser registrado: 4.1-Abrir uma demanda de defeito conforme instruções contidas no DA.0002; 5-Caso os testes de aceite obtenham resultados satisfatórios: 5.1-Emitir o Termo de Aceite para assinatura por parte do Cliente; 5.2-Alterar o status da demanda de Testes de Aceite para “Resolvido” lançando o esforço utilizado para o cumprimento da atividade.		
Saídas	**Critérios:** - Testes de Aceite concluídos; - Não haver defeitos críticos em aberto; - Demanda de execução de aceite com status “Resolvido”.		
	Produtos: - Termo de Aceite; - Registros de defeitos.		

Tabela 3.8: *Descrição da prática encerrar teste de aceite*

Objetivos	Elaborar um relatório conclusivo sobre a execução dos testes de aceite no release disponibilizado para testes, a fim de obter um laudo da qualidade da aplicação.		
Participantes	**Responsável:** Gerente de Teste	**Colaborador:** Analista de Teste	**Aprovação:**
Entradas	**Critérios:** - Execução dos Testes de Aceite concluída; - Demanda de Teste de Aceite com status “Resolvido”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - Termo de Aceite; - Registros de defeitos.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0003 no repositório de templates; 3-Validar e consolidar os resultados obtidos no RQ.0003 na área destinada a Teste de Aceite; 3.1-Caso necessário, atribuir a demanda de volta ao Analista de Teste responsável para ajustes. 4-Armazenar o RQ.0003 no diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Fechar as demandas de Teste de Aceite lançando o esforço utilizado na execução da tarefa.		
Saídas	**Critérios:** - Consolidação de resultados da Execução de Testes de Aceite concluída; - Demandas de Teste de Aceite com status “Fechado”.		
	Produtos: - RQ.0003 – Relatório Consolidado de Teste.		

Tabela 3.9: *Descrição da prática preparar massa de teste*

Objetivos	Preparar a massa de testes necessária para execução automatizada do testes de regressão ou desempenho.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Testes manuais concluídos; - Demanda inserida no escopo dos testes de regressão ou desempenho; - Demanda Teste de Regressão com status “Aberto”; - Demanda Teste de desempenho com status “Aberto”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - DA.0003 – Diretrizes para Massa de Teste.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Alterar o status da demanda Teste de Regressão ou desempenho para “Em andamento”; 3-Recuperar o modelo de RQ.0004 no devido repositório de templates; 4-Elaborar a massa de teste necessária no RQ.0004 conforme instruções citadas no DA.0003, sendo que: 4.1-Regressão: considerar os Cenários Funcionais; 4.2-Desempenho: considerar os Cenários de desempenho. 5-Armazenar o RQ.0004 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 6-Caso haja necessidade de editar os scripts de regressão ou performance: 6.1-Encaminhar a demanda de Teste de Regressão ou Performance para edição de seu script, lançando o devido esforço destinado à execução da tarefa. 7-Caso contrário: 7.1-Encaminhar a demanda de Teste de Regressão ou Performance para Execução de seu script, lançando o devido esforço destinado à execução da tarefa.		
Saídas	**Critérios:** - Massa de teste devidamente elaborada; - Demanda de Teste de Regressão ou Performance encaminhado para “Edição” ou “Execução”.		
	Produtos: - RQ.0004 – Massa de Teste.		

Tabela 3.10: *Descrição da prática manter script de regressão*

Objetivos	Preparar o script de testes necessário para execução automatizada do testes de regressão.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Massa de teste elaborada; - Cenário de teste especificado; - Demanda Teste de Regressão com status “Em andamento”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - DA.0004 – Diretrizes para Script de Teste; - RQ.0002 – Cenários de Testes; - RQ.0004 – Massa de Teste.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0005 no devido repositório de templates; 3-Elaborar o script de regressão necessário no RQ.0005 na ferramenta de automatização funcional (considerar apenas os Cenários Funcionais no RQ.0002), seguindo as instruções citadas no DA.0004; 4-Armazenar o RQ.0005 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Lançar o devido esforço utilizado na execução na demanda de Teste de Regressão; 6-Encaminhar a demanda de Teste de Regressão para execução de seu script.		
Saídas	**Critérios:** - Script de regressão funcionando; - Demanda de Teste de Regressão encaminhada para “Execução”.		
	Produtos: - RQ.0005 – Script de Teste.		

Tabela 3.11: *Descrição da prática executar teste de regressão*

Objetivos	Realizar os testes de regressão para a nova versão da aplicação.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Ambiente de teste disponível com nova versão da aplicação; - Demanda de Teste de Regressão com status “Em andamento”		
	Insumos: - DA.0002 – Padrão para Registro de Defeito; - RQ.0004 – Massa de Teste; - RQ.0005 – Script de Teste.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Executar o script de testes de regressão a partir da ferramenta de automatização funcional, sendo: 2.1-Imediato: Acompanhar a execução do script de teste de regressão; 2.2-Agendado: Agendar a execução dos testes de regressão para um horário mais apropriado 3-Coletar resultados dos testes de regressão sobre a aplicação e reportar os resultados obtidos na ferramenta de gestão de testes; 4-Havendo algum defeito a ser registrado: 4.1-Abrir uma demanda de defeito conforme instruções contidas no DA.0002; 5-Resolver a demanda de Teste de Regressão lançando o devido esforço destinado à execução da tarefa.		
Saídas	**Critérios:** - Execução de testes de regressão concluída e resultados devidamente lançados na ferramenta de gestão de testes; - Não haver defeitos críticos em aberto; - Demanda de Teste de Regressão com status “Resolvido”.		
	Produtos: - Registros de defeitos; - Log de execução; - Evidências de testes.		

Tabela 3.12: *Descrição da prática encerrar teste de regressão*

Objetivos	Elaborar um relatório conclusivo sobre a execução dos testes de regressão no release disponibilizado para testes, a fim de obter um laudo da qualidade da aplicação.		
Participantes	**Responsável:** Gerente de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Execução dos Testes de Regressão concluída; - Demanda de Teste de Regressão com status “Resolvida”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - Registros de defeitos.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0003 no devido repositório de templates; 3-Validar e consolidar os resultados obtidos no RQ.0003 na área destinada a Teste de Regressão; 3.1-Caso necessário, atribuir a demanda de volta ao Analista de Teste responsável para devidos ajustes. 4-Armazenar o RQ.0003 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Fechar as demandas de Teste de Regressão lançando o devido esforço utilizado na execução da tarefa.		
Saídas	**Critérios:** - Consolidação de resultados da Execução de Testes concluída; - Demandas de Teste de Regressão com status “Fechado”.		
	Produtos: - RQ.0003 – Relatório Consolidado de Teste.		

Tabela 3.13: *Descrição da prática manter script de desempenho*

Objetivos	Preparar os scripts de testes necessários para execução automatizada dos testes de desempenho.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Massa de teste elaborada; - Cenário de teste especificado; - Demanda Teste de desempenho com status “Em andamento”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - DA.0004 – Padrões de Script de Teste; - RQ.0002 – Cenários de Testes; - RQ.0004 – Massa de Teste.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0005 no repositório de templates; 3-Elaborar o script de desempenho necessário no RQ.0005 na ferramenta de automatização de desempenho (considerar apenas os Cenários de desempenho no RQ.0002), seguindo as instruções citadas no DA.0004; 4-Armazenar o RQ.0005 no diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Lançar o esforço utilizado na execução na demanda de Teste de desempenho; 6-Encaminhar a demanda de Teste de desempenho para execução de seu script.		
Saídas	**Critérios:** - Script de desempenho funcionando; - Demanda de Teste de desempenho encaminhada para “Execução”.		
	Produtos: - RQ.0005 – Script de Teste.		

Tabela 3.14: *Descrição da prática executar teste de desempenho*

Objetivos	Realizar os testes de desempenho para a nova versão da aplicação.		
Participantes	**Responsável:** Analista de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Ambiente de teste disponível com nova versão da aplicação; - Demanda de Teste de desempenho com status “Em andamento”.		
	Insumos: - DA.0002 – Padrão para Registro de Defeito; - RQ.0004 – Massa de Teste; - RQ.0005 – Script de Teste.		
Ações	1-Havendo alguma inconformidade na abertura da demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Executar o script de testes de desempenho a partir da ferramenta de automatização de desempenho, sendo: 2.1-Imediato: Acompanhar a execução do script de teste de desempenho; 2.2-Agendado: Agendar a execução dos testes de desempenho para um horário mais apropriado 3-Coletar resultados dos testes de desempenho sobre a aplicação e reportar os resultados obtidos na ferramenta de gestão de testes; 4-Havendo algum defeito a ser registrado: 4.1-Abrir uma demanda de defeito conforme instruções contidas no DA.0002; 5-Resolver a demanda de Teste de desempenho lançando o devido esforço destinado à execução da tarefa.		
Saídas	**Critérios:** - Execução de testes de desempenho concluída e resultados devidamente lançados na ferramenta de gestão de testes; - Não haver defeitos críticos em aberto; - Demanda de Teste de desempenho com status “Resolvido”.		
	Produtos: - Evidências de testes; - Log de execução; - Registros de defeitos.		

Tabela 3.15: *Descrição da prátcia encerrar teste de desempenho*

Objetivos	Elaborar um relatório conclusivo sobre a execução dos testes de desempenho na versão disponibilizada para testes, a fim de obter um laudo da qualidade da aplicação.		
Participantes	**Responsável:** Gerente de Teste	**Colaborador:**	**Aprovação:**
Entradas	**Critérios:** - Execução dos Testes de desempenho concluída; - Demanda de Teste de desempenho com status “Resolvida”.		
	Insumos: - DA.0001 – Padrões de Nomenclatura; - Registros de defeitos.		
Ações	1-Havendo alguma inconformidade na demanda: 1.1-Atribuir a demanda de volta ao solicitante informando as necessidades de ajustes. 2-Recuperar o modelo de RQ.0003 no devido repositório de templates; 3-Validar e consolidar os resultados obtidos no RQ.0003 na área destinada a Teste de desempenho; 3.1-Caso necessário, atribuir a demanda de volta ao Analista de Teste responsável para devidos ajustes. 4-Armazenar o RQ.0003 no devido diretório de testes existente no repositório do release, fazendo uso dos padrões de nomenclatura citados no DA.0001; 5-Fechar as demandas de Teste de desempenho lançando o devido esforço utilizado na execução da tarefa.		
Saídas	**Critérios:** - Consolidação de resultados da Execução de Testes concluída; - Demandas de Teste de desempenho com status “Fechado”.		
	Produtos: - RQ.0003 – Relatório Consolidado de Teste.		

CAPÍTULO 4

Conclusão

Como enunciado anteriormente nesse documento a obtenção da qualidade em produtos de software é influenciada por diversos fatores como escopo do produto, tempo disponível para execução do projeto, grau de conhecimento da equipe técnica, dificuldade do negócio, dentre vários outros. Tendo observado esses pontos é possível explanar sobre as profundas contribuições que o teste de software se propõe a promover no âmbito da qualidade.

Apesar de existirem inúmeras técnicas, políticas e metodologias na área de teste muitas dessas que são adequadas para grandes empresas e fábricas de software não se encaixam nas necessidades das micro e pequenas empresas. Isso ocorre pois as empresas de menor porte possuem uma realidade diferente das demais, contando com um número restrito de recursos – disponibilidade de profissionais, orçamento, dentre outros.

No panorama restrito das micro e pequenas empresas a utilização consciente dos recursos faz toda a diferença no valor agregado final de seus produtos. Não obstante, o uso correto de processos de teste de software e ferramentas que o apoiem de forma integrada tendem a contribuir de maneira benéfica para a qualidade e nesse valor final. Dessa forma tanto o *framework* de processo quanto a integração das ferramentas de teste foram desenvolvidos pensando nas realidades dessas empresas de forma que seja o mais próximo possível de suas necessidades, cobrindo uma fatia importante de conhecimento antes não massivamente explorado.

A implantação de teste de software nas empresas depende então não somente de uma postura otimista da parte das gerências, pois exige grande empenho e uma mudança geral na cultura de como o software é produzido, mas também de uma iniciativa pessoal de cada profissional das equipes envolvidas. Sendo assim, a visão de melhoria causada pelo teste deve ser pensado como um investimento a médio ou longo prazo, mas que certamente recompensará os esforços aplicados.

Bibliografia

BROOKS JR., F. P. No silver bullet essence and accidents of software engineering. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 20, n. 4, p. 10–19, abr. 1987. ISSN 0018-9162. Disponível em: <http://dx.doi.org/10.1109/MC.1987-.1663532>.

BURNSTEIN, I. *Practical Software Testing*. 1st. ed. [S.l.]: Springer, 2003.

BURNSTEIN, I.; SUWANASSART, T.; CARLSON, R. Developing a Testing Maturity Model for Software Test Process Evaluation and Improvement. In: *International Test Conference*. [S.l.]: IEEE, 1996.

CERCOMP-UFG. *Manual de produção de software do Cercomp*. Goiânia-Goiás, Brasil, 2010.

IEEE. *IEEE Std 610: Computer dictionary a compilation of IEEE Standard Computer Glossaries*. [S.l.]: Institute of Eletrical and Eletronics Engineers, 1990.

JACOBS, J. C.; TRIENEKENS, J. J. M. Towards a Metrics Based Verification and Validation Maturity Model. In: *Proceedings of the 10th International Workshop on Software Tecnology and Engineering Practice*. [S.l.]: IEEE, 2002.

MULLER, D. F. T. *Certified Tester Foundation Level Syllabus*. [S.l.]: Comissão Internacional para Qualificação de Teste de Software, 2011.

ROCHA, A. R. C.; MACHADO, C. Ângela F. *MPS.BR - Melhoria de Processo do Software Brasileiro - Guia Geral*. [S.l.]: SOFTEX, 2011.

SOFTEXRECIFE; RIOSOFT. *Guia de referência do modelo - MPT.Br*. [S.l.]: SOFTEXRECIFE, 2011.

VEENENDAAL, E. van. *Test Maturity Model integration*. 1st. ed. [S.l.]: TMMi Foundation, 2012.

YU, Y. T.; LAU, M. F. A comparison of mc/dc, mumcut and several other coverage criteria for logical decisions. *J. Syst. Softw.*, Elsevier Science Inc., New York, NY, USA, v. 79, p.

577–590, May 2006. ISSN 0164-1212. Disponível em: <http://dx.doi.org/10.1016/j.jss-.2005.05.030>.

APÊNDICE A

Glossário

Alguns termos mencionados neste trabalho são apresentados na Tabela A.1 seguido do respectivo nome (ou sigla) e da definição.

Tabela A.1: *Glossário*

Termo	**Definição**
Analista de teste	Responsável por modelar os cenários de testes e gerar os scripts de testes automatizados quando necessário.
Área de processo	Agrupamento de práticas relacionadas que, quando implementadas em conjunto, satisfazem um determinado objetivo (SOFTEXRECIFE; RIOSOFT, 2011).
Linha-base (*Baseline*)	Uma versão formalmente aprovada de um conjunto de itens de configuração e que serve de base para trabalhos futuros.
Caso de teste	Um conjunto de entradas de teste, condições de execução e resultados esperados desenvolvidos para atender um objetivo específico, como por exemplo exercitar um caminho do programa em particular ou para verificar inconsistência com um requisito específico (IEEE, 1990).
Critério de teste	Se trata de um critério que um sistema ou componente deve encontrar e passar para exercitar um teste específico (IEEE, 1990).
Decisão (*Decision*)	É uma expressão booleana composta de condições com zero ou mais operadores booleanos (YU; LAU, 2006).
Decision Coverage (DC)	Cada decisão tem que assumir todos os possíveis valores pelo menos uma vez (YU; LAU, 2006).
Gerente de projetos	Responsável por assegurar que o objetivo do projeto seja cumprido, com qualidade e dentro do período definido.
Continua na próxima página...	

Tabela A.1: *Glossário*

Termo	**Definição**
Gerente de teste	Responsável em definir e acompanhar os projetos de testes.
Processo	Conjunto de atividades que transformam entradas em saídas, a fim de atingir um propósito bem determinado (CERCOMP-UFG, 2010).
Projeto	Empreendimento executado para fornecer um produto ou serviço único (CERCOMP-UFG, 2010).
Release	Versão da configuração de software que constitui uma *baseline* de produto e que será ou foi entregue ao patrocinador.
Técnica de teste	O processo de operação de um sistema ou componente durante condições específicas, observações ou registro de resultados e avaliação de alguns aspectos do sistema ou componente (IEEE, 1990).
Testador	Responsável por executar os testes e relatar os resultados obtidos.

APÊNDICE B

Template do Plano de Teste

B.1 Visão Geral do Projeto

<Descrever em linhas gerais o software a ser testado, comunicando o propósito da aplicação, e a importância do projeto para todas as pessoas envolvidas.>

B.1.1 Papéis e Responsabilidades

<Descrever na Tabela B.1 os papéis existentes com suas devidas responsabilidades.>

Tabela B.1: *Papéis e Responsabilidades do Projeto*

Papel	Responsabilidade

B.2 Equipe e Infra-estrutura

<Detalhar a equipe e a infra-estrutura utilizada pelo processo de teste, incluindo: pessoal, equipamentos, ferramentas, software de apoio, materiais, etc.>

B.2.1 Planejamento da Alocação de Pessoal

<Expor informações referentes a equipe de profissionais alocada em tempo integral ou parcial para o teste do sistema, indicando o número de recursos necessários para cada um dos papéis definidos na Seção B.1.1. Caso o projeto tenha sido realizado em fases ou iterações e para cada uma delas houve diferenças no número de profissionais alocados, é importante incluir estas informações por fase/iteração na Tabela B.2.>

Tabela B.2: *Planejamento de Alocação de Pessoal*

Papel	Quantidade	Tempo de Dedicação Semanal

B.2.2 Softwares/Equipamentos

<Opcional. São relacionados na Tabela B.3 os softwares/equipamentos necessários para a equipe de teste do projeto, sua finalidade (por exemplo, realização de testes de aceitação), a quantidade necessária e o período de utilização do software/equipamento.>

Tabela B.3: *Relação de Softwares/Equipamentos Necessários*

Descrição	Finalidade	Quantidade	Período

B.3 Artefatos

<Mencionar os artefatos de entrada (conforme Tabela B.4) que são necessários e imprescindíveis para que as atividades de testes possam ser iniciadas. Também deverão ser informados na Tabela B.5 os artefatos que serão gerados para que sejam evidenciadas as execuções dos testes realizados.>

Tabela B.4: *Lista de Artefatos de Entrada*

Artefato de Entrada	Responsável	Observações

Tabela B.5: *Lista de Artefatos de Saída*

Artefato de Saída	Responsável	Observações

B.4 Treinamentos

<Opcional. Descrever como será conduzida a capacitação dos profissionais para realização das atividades e utilização das ferramentas adotadas no desenvolvimento do projeto. Adicionar na Tabela B.6 informações sobre os treinamentos exclusivos para o projeto. Ex.: tecnologias específicas, sistemas legados, negócio etc.>

Tabela B.6: *Relação de Treinamentos/Cursos*

Descrição	Quantidade	Carga Horária	Período	Instrutor

B.5 Acompanhamento do Projeto

<Deverão ser relacionados os momentos para realização das atividades de verificação do projeto sob o escopo do teste, as quais poderão ser feitas pelos envolvidos no projeto.>

B.5.1 Acordo de Nível de Serviço

<Estabelecer quais serão os acordos de nível de serviço entre a Equipe de Teste e o projeto de desenvolvimento do sistema, por meio de critérios objetivos e factíveis de avaliação. Os intermediadores entre esses acordos serão o Gerente de Teste e o Gerente do Projeto, sendo os níveis de serviço avaliados durante as atividades de acompanhamento do projeto mencionadas na Seção B.5.>

B.6 Cronograma

<No cronograma devem constar as atividades, marcos de projeto (*milestones*), dependências e recursos humanos alocados. É importante que o *baseline* seja registrado, a fim de garantir dados para análise de planejado *versus* realizado.>

B.6.1 Marcos Significativos do Teste

<Descrever os importantes marcos do projeto (incluindo as datas de início e fim do projeto), bem como os artefatos que serão entregues ao cliente nestes marcos, quando aplicável. Apenas marcos significativos devem ser listados, ou seja, aqueles que contribuirão para a medição do desempenho do projeto. Ex. reuniões de revisão, disponibilização de protótipos, realização de testes de aceitação, etc.

É possível inserir uma visão do cronograma do projeto nesta seção, destacando apenas os marcos importantes e suas respectivas datas, em substituição à Tabela B.7.>

Tabela B.7: *Marcos Significativos do Projeto*

Marco	Artefatos	Data

B.7 Estratégia de Testes

<Apresentar os níveis e tipos de testes a serem realizados no projeto. Na Tabela B.8 deverão ser mencionadas as categorias de testes existentes e que serão contempladas no projeto.>

Tabela B.8: *Categorias de Testes do Projeto*

Categorias de Teste	Objetivo

B.8 Critério de Finalização

B.8.1 Critério de Conclusão de Testes

<Para considerar o término dos trabalhos referentes a essa fase de testes, é necessário estabelecer critérios de finalização dos objetivos, visando tornar claro qual será a regra para considerar o software de qualidade.>

B.8.2 Critério de Aceitação

<Definir com o usuário/cliente quais os critérios de aceitação são necessários para concluir a fase de homologação, e liberação do produto para produção.>

B.9 Gerência de Riscos

<Utilizar o modelo de Planilha de Riscos, criar novo documento para controlar os riscos dos testes. Esse documento apresenta a lista de riscos identificados, seus impactos e probabilidades de ocorrência, ações de contingência e/ou mitigação planejadas e seus respectivos responsáveis, sob o aspecto teste. Todo o acompanhamento dos riscos do projeto relacionados a teste (riscos previamente identificados e riscos surgidos no decorrer do andamento do projeto) será registrado no plano supracitado.>

B.10 Termo de Aceite

<Os profissionais abaixo declaram estar cientes de todos os itens descritos e acordados neste documento, bem como da viabilidade da dedicação de recursos, esforços e responsabilidades aqui definidos:>

_____

Representante da Gerência de Projeto

_____

Representante da Área de Qualidade

_____

Representante da Área de Negócios

APÊNDICE C

Template de Cenários de Teste

C.1 Funcionalidades

<Descrever em linhas gerais a(s) necessidade(s) da funcionalidade.>

C.1.1 Necessidade *n*

- Como um <usuário/ator>
- Eu quero <meta a ser alcançada>
- De modo que <a razão para alcançar a meta>

C.2 Cenários Funcionais

<Descrever todos os cenários previstos para execução dos testes.>

C.2.1 Cenários *n*

<Descrever detalhadamente o cenário a ser considerado para a execução de testes.>

- Descrição do Cenário: <descrição do teste>
- Dado que <procedimentos/um estado conhecido>
- Quando <um determinado evento ocorre>
- Então <resultado esperado>

C.3 Cenários de Desempenho

<Descrever todos os cenários previstos para execução dos testes de desempenho.>

C.3.1 Cenários *n*

<Descrever detalhadamente o cenário a ser considerado para a execução de testes de performance.>

- Descrição do Cenário: <descrição do teste>
- Dado que <procedimentos/um estado conhecido>
- Quando <um determinado evento ocorre>
- Então <resultado esperado>

APÊNDICE D

Template do Relatório de Consolidação de Teste

D.1 Escopo dos Testes

<Descrever as funcionalidades com suas devidas demandas que foram envolvidas nas atividades de Verificação e Validação nas Tabelas D.1, D.2, D.3, D.4 e D.5>

Tabela D.1: *Teste de Requisitos*

Teste de Requisitos		
Funcionalidade	Módulo	Demandas

Tabela D.2: *Teste Funcionais*

Teste Funcionais		
Funcionalidade	Módulo	Demandas

Tabela D.3: *Teste de Regressão*

Teste de Regressão		
Funcionalidade	Módulo	Demandas

Tabela D.4: *Teste de Performance*

Teste de Performance		
Funcionalidade	Módulo	Demandas

Tabela D.5: *Teste de Aceite*

Teste de Aceite		
Funcionalidade	Módulo	Demandas

D.2 Resultados Obtidos

<Descrever os resultados obtidos durante a realização dos diversos testes a serem executados durante o ciclo de vida da *release* nas Tabelas D.6, D.7, D.8, D.9 e D.10.>

Tabela D.6: *Defeitos Encontrados Durante a Realização do Teste de Requisitos*

Defeitos - Teste de Requisitos			
Criticidade	Corrigidos	Não Corrigidos	Total
Alta			
Média			
Baixa			

Tabela D.7: *Defeitos Encontrados Durante a Realização do Teste Funcional*

Defeitos - Teste de Funcional			
Criticidade	Corrigidos	Não Corrigidos	Total
Alta			
Média			
Baixa			

Tabela D.8: *Defeitos Encontrados Durante a Realização do Teste de Regressão*

Defeitos - Teste de Regressão			
Criticidade	Corrigidos	Não Corrigidos	Total
Alta			
Média			
Baixa			

Tabela D.9: *Defeitos Encontrados Durante a Realização do Teste de Performance*

Defeitos - Teste de Performance			
Criticidade	Corrigidos	Não Corrigidos	Total
Alta			
Média			
Baixa			

Tabela D.10: *Defeitos Encontrados Durante a Realização do Teste de Aceite*

Defeitos - Teste de Aceite			
Criticidade	Corrigidos	Não Corrigidos	Total
Alta			
Média			
Baixa			