Estudo de Caso - Amsoft

O processo de teste de software tem sido procurado por muitas empresas de desenvolvimento de software para que seus produtos atinjam níveis de qualidade desejáveis. Este artigo apresenta uma série de processos de teste (Ideal, CenpRA, 3Px3E) que podem ser implantados em organizações.

O conteúdo apresentado faz parte de um estudo que foi realizado com objetivo de implantar uma metodologia de testes em uma empresa de desenvolvimento de software e é útil para aqueles que tenham interesse na implantação de uma abordagem de testes em sua organização e tenham interesse em conhecer iniciativas similares que possam servir de ponto de partida para suas atividades.

A informação faz parte do dia a dia das pessoas. Para cada decisão a ser tomada, seja no trabalho, na escola ou na vida pessoal, temos uma série de informações a serem consultadas. Para o gerenciamento delas são utilizados os sistemas de informação. Estes são mecanismos de gestão que atuam como condutores das informações facilitando, agilizando e aperfeiçoando o processo decisório nas empresas.

A engenharia do software determina padrões de sistemas de informação, nela se encontram os princípios de qualidade dos sistemas.

Atualmente existem modelos de qualidade baseados na engenharia que ajudam as empresas a estabelecer e seguir padrões. Podemos citar como exemplos destes modelos o CMM - Capability Maturity Model, a ISO - International Organization for Standardization e o MPS.BR - Processo de Melhoria do Software Brasileiro. O objetivo principal dos modelos de qualidade é fazer com que toda a equipe esteja trabalhando no mesmo foco.

Além dos modelos, é importante que as organizações adotem o teste de software. Esta fase do processo da engenharia do software é importante, pois avalia se o produto atende aos requisitos especificados. É um método destrutivo, que é executado com o intuito de encontrar os erros e corrigi-los antes da versão final ser liberada ao cliente.

As metodologias de teste de software especificam padrões sobre como os mesmos devem ser executados, como criar a documentação necessária, enfim, projetar da melhor maneira o processo.

Diante das colocações, observa-se que as empresas, principalmente as de pequeno e médio porte, sentem dificuldade de associar metodologias de teste de software a seus processos de desenvolvimento. A dificuldade se dá muitas vezes pelos recursos financeiros e também de pessoal serem limitados.

A empresa AmSoft, desenvolvedora de sistemas de informação gerenciais, localizada na cidade de Pato Branco, está enfrentando este mesmo impasse. Sendo assim, foi proposto investigar as metodologias de teste de software mais utilizadas atualmente, selecionar entre elas a que se destaca, usando critérios de seleção estabelecidos pela empresa AmSoft, na qual será realizado o estudo de caso. Em seguida, implantar a metodologia selecionada descrevendo e analisando os resultados obtidos, podendo assim, justificar a importância de sua utilização.

Muitas Softhouses, classificadas como micro, pequenas e médias empresas, estão sentindo a necessidade de melhoria em seus processos internos. Como alternativa estão recorrendo a

metodologias, entre elas de teste de software. Assim, têm surgido entre as mesmas o seguinte questionamento: uma metodologia de teste de software pode auxiliar na organização dos processos internos da empresa, fazendo com que isso reflita em qualidade no produto, satisfação por parte dos clientes e destaque no mercado competitivo, o que justificaria a importância de sua utilização?

A empresa AmSoft, foco principal do estudo de caso a ser apresentado, enfrenta este mesmo impasse. Diante disso, observou-se a necessidade de aplicação de uma metodologia de testes que seja adaptável a esse nicho de mercado.

A utilização desse tipo de metodologia pode evitar muitos problemas tais como: a redução de custos com retrabalho, manutenção, leva a minimizar erros entregues com a versão final ao cliente, o que reflete em um aumento do nível de confiança por parte dos mesmos. Com mais tempo disponível, a equipe pode investir em melhorias e novos planejamentos.

A análise também pode servir de referência para outras empresas que se encontram na mesma situação e precisam efetuar melhorias nesse aspecto. Possibilita também um conhecimento mais detalhado sobre qualidade, testes e metodologias de teste de software.

Informação

Dentro do conceito de sistemas de informação, informação é o resultado do processamento dos dados de entrada. Na **Figura 1** está representado o processo de transformação dos dados em informação. Estes são coletados, passam pela fase de processamento ou transformação e em seguida são devolvidos sendo tratados já como informação.

Figura 1. Transformação de dados em informação

A informação é o dado que foi processado e armazenado de forma compreensível para seu receptor e que apresenta valor real percebido para suas decisões correntes ou prospectivas.

Os dados podem ser definidos como sucessões de fatos brutos, que não foram organizados, processados, relacionados, avaliados ou interpretados, representando apenas partes isoladas de eventos, situações ou ocorrências. Constituem as unidades básicas a partir das quais informações poderão ser elaboradas ou obtidas.

Processamento são as transformações, alterações, relacionamentos, agrupamentos, pelos quais os dados passam para serem transformados em informação.

Sistemas de informação

Os sistemas de informação são mecanismos de apoio à gestão, desenvolvidos com base na tecnologia de informação e com suporte da informática para atuar como condutores das informações que visam facilitar, agilizar e aperfeiçoar o processo decisório nas organizações.

Eles compreendem um conjunto de recursos humanos, materiais, tecnológicos e financeiros, agregados segundo uma sequência lógica para o processamento dos dados e a correspondente tradução em informações. A **Figura 2** mostra um modelo de sistema de informação.

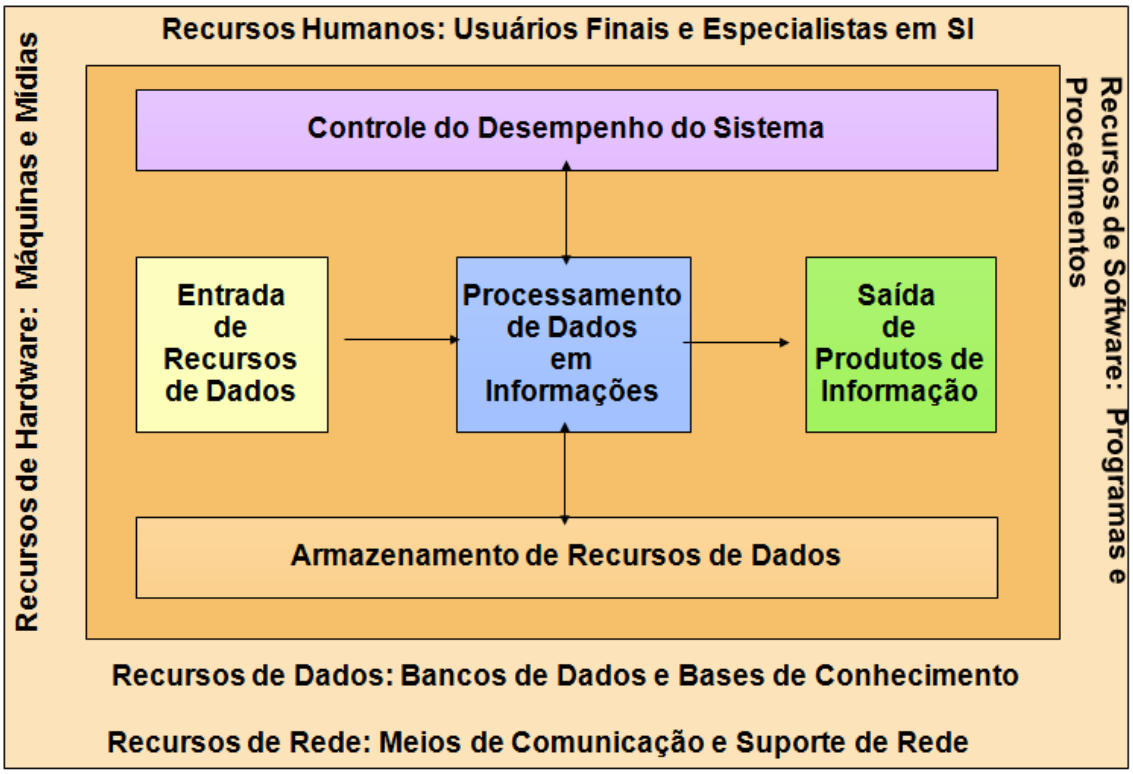

Figura 2. Modelo de Sistema de Informação

O objetivo desses sistemas é gerar informação para serem usadas no auxílio da tomada de decisões de empresas e organizações, onde as mesmas terão condições de reagir às mudanças constantes do mercado. Dentro deste conceito, temos exemplos de sistemas utilizados atualmente: SIG - Sistemas de Informação Gerencial e ERP - Enterprise Resource Planning do português Planejamento de Recursos Empresariais.

Sistema de informação gerencial pode ser definido como qualquer sistema que produza posições atualizadas no âmbito corporativo, resultado da integração de vários grupos de sistemas de informação que utilizam recursos de consolidação e interligação de entidades dentro de uma organização.

O propósito básico de um SIG é ajudar a empresa a alcançar suas metas, fornecendo a seus gerentes detalhes sobre as operações regulares da organização de forma que possam controlar, organizar e planejar com mais efetividade e com maior eficiência.

Na **Figura 3** podemos observar o conceito de um sistema de informação gerencial. O programa de aplicação juntamente com o Gerenciador de Banco de Dados – SGBD estão conectados ao banco de dados da empresa no qual são armazenadas as informações. Em seguida disponibilizam-nas na forma de relatórios e gráficos ao gerente, auxiliando-o na tomada de decisões.

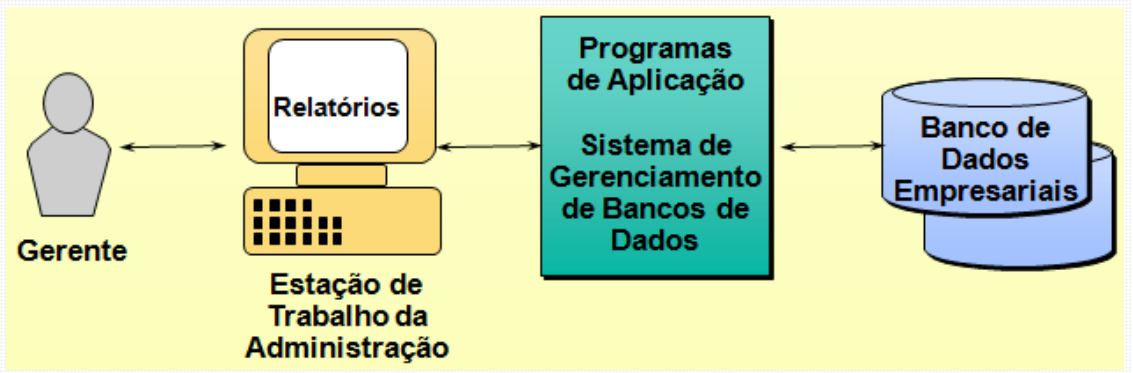

Figura 3. Conceito de Sistemas de Informação Gerencial

Os ERP – Enterprise Resource Planning surgiram da necessidade de se controlar todos os principais processos da organização em um só software. Buscavam-se benefícios como eficiência, incremento da qualidade, da produtividade e lucratividade.

O principal objetivo dos sistemas ERP é integrar todos os departamentos e funções da empresa em um sistema unificado de informática, com capacidade de atender todas as necessidades da organização.

A primeira geração dos ERP's foi fundamental para pequenas e médias empresas concentrar-se em processos de negócio. Eles as conduziam a uma nova maneira de pensar e expandir suas operações com uma melhor gestão.

Como a ideia principal era automatizar processos administrativos e sua capacidade estava quase esgotada, no final da década de 90 surgiu a segunda geração dos ERP's. O objetivo desta segunda geração é alavancar os sistemas existentes de maneira a aumentar a eficiência no encaminhamento das transações, melhorar o processo decisório e reforçar ainda mais os métodos de negócio.

Na **Figura 4** podemos observar a junção de vários recursos: Logística, Gestão, Vendas, Financeiro e Produção, que antes eram sistemas separados, todos estes unidos em um único sistema: o ERP.

Figura 4. Conceito de Sistemas ERP

Engenharia de Software

Diversos governos de países em desenvolvimento têm mostrado preocupação com a questão da informatização de seus órgãos, empresas e da sociedade como um todo. Isso tem despertado nas organizações a procura por ferramentas e metodologias que auxiliem nos seus processos.

Software é um conjunto de instruções que são executadas com objetivo de produzir funções e desempenhos desejados. Também pode ser definido como estruturas de dados que permitem aos programas manipular informações. Ou ainda, documentos que descrevem operação e uso dos programas.

O software tem se tornado importante para as empresas, como um diferencial de desempenho e controle, o que proporciona competitividade diante da concorrência. Sobre este referencial, as empresas de desenvolvimento se mostram cada vez mais preocupadas com a engenharia dos mesmos, com intuito de atribuir qualidade aos produtos desenvolvidos.

Com o intuito de melhorar a qualidade dos softwares em geral e aumentar a produtividade no desenvolvimento de tais produtos, surgiu a engenharia de software. Esta é um conjunto de três elementos: métodos, ferramentas e procedimentos que possibilitam ao gerente controlar o processo de desenvolvimento e, ao profissional, oferece base para construção de um produto com alta qualidade, de forma produtiva.

Os métodos indicam como fazer para construir o produto. Envolvem: planejamento, estimativa de projeto, análise de requisitos, projeto da estrutura de dados, arquitetura de programa, algoritmo de processamento, codificação, teste e manutenção.

As ferramentas proporcionam apoio automatizado ou semi-automatizado aos métodos, para cada um existem ferramentas de sustentação.

Os procedimentos são a ligação entre os métodos e as ferramentas. Definem a sequência de aplicação, os produtos a serem entregues, os controles que asseguram a qualidade e a coordenação de mudanças e os marcos de referência para avaliação do progresso.

Sendo assim, a engenharia de software pode ser definida como um conjunto de princípios para desenvolvimento de um produto com alto nível de qualidade e confiança, que funcione de maneira eficiente. Para isso, a qualidade se torna o marco principal.

Qualidade de Software

O principal objetivo da engenharia do software é auxiliar na construção de produtos de qualidade. Qualidade é atender perfeitamente, de forma confiável (sem defeitos), acessível (de baixo custo), segura e no tempo certo as necessidades ou requisitos do cliente.

A qualidade também pode ser entendida como sendo a conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido.

A qualidade é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir a conformidade de processos e produtos, prevenindo e eliminando defeitos.

Se no contexto da engenharia do software qualidade significa satisfazer os requisitos especificados e as necessidades e expectativas do cliente, assegurar esta “qualidade” nem sempre é tão simples quanto parece.

Geralmente, no meio do caminho são encontrados requisitos implícitos, às vezes com representações variadas ou até mesmo incompletos, isso prejudica o produto, causa retrabalho e afeta no custo e prazos de entrega.

Para evitar estes impasses torna-se necessário incorporar métodos e conceitos como “Planejamento”, “Controle” e “Garantia de Qualidade”.

Planejamento, Controle e Garantia de Qualidade

No planejamento da qualidade serão definidas as atividades de avaliação que serão executas ao longo do projeto. O que será avaliado, quando vai ocorrer esta avaliação, como será executado e por quem.

A garantia da qualidade avalia a aderência das atividades executadas e dos produtos de trabalho, de acordo com padrões, processos, procedimentos e requisitos estabelecidos e aplicáveis. Ela assegura que a qualidade que foi planejada não seja comprometida. Busca identificar desvios o quanto antes, eliminando possíveis retrabalhos e melhorando custos e prazos.

O controle de qualidade pode ser definido como um método de comparação entre o produto e os requisitos apresentados no projeto. É feita uma verificação determinando se o produto está dentro dos níveis aceitáveis. Diferentes níveis de teste podem ser usados para esse controle de qualidade.

As diferenças entre a garantia e o controle estão apresentadas na **Tabela 1**.

Garantia de Qualidade	Controle de Qualidade
■ Foco: Garantir que o projeto emprega todos os processos e padrões necessários para atender aos requisitos. ■ Forma mais usual: auditorias de processo e de produto orientados por *checklists*. ■ Utiliza métodos, procedimentos e padrões para comparar previsto com realizado. ■ Assegura que o processo empregado é definido e apropriado. ■ É orientada a processo, visando a prevenção de defeitos. ■ Cuida da monitoração e melhoria dos processos e padrões empregados. ■ Assegura que se faz de maneira correta.	■ Foco: Descobrir defeitos em produtos de trabalho gerados ao longo do projeto e eliminar suas causas. ■ Forma mais usual: testes diversos e revisões por pares. ■ Utiliza casos de teste, *checklists* e revisões para comparar o esperado com o obtido. ■ Assegura que os produtos de trabalho gerados estão consistentes e alinhados. ■ É orientado a produto, visando à detecção e correção de defeitos. ■ Cuida da monitoração e da consistência dos produtos em relação aos requisitos e à utilização. ■ Assegura que se faz as coisas certas.

Tabela 1. Diferenças entre "garantia" e "controle" da qualidade

Modelos de Qualidade

Os modelos de qualidade implicam em padrões e procedimentos de trabalho que apontam uma sequência de fases com objetivos específicos, que adaptam às empresas processos de acordo com as necessidades do mercado.

As desenvolvedoras de software podem optar por desenvolver um modelo próprio de acordo com a necessidade do mercado que atuam, ou seguir modelos já existentes como o ISO - International Organization for Standardization, CMM - Capability Maturity Model e no Brasil o MPS.BR.

Os modelos buscam o trabalho em grupo, os objetivos comuns, ou seja, a colaboração de toda a equipe sincronizada em um único foco para melhor desempenho dos projetos. Assim, é necessário criar estratégias, com processos de tomada de decisão, gerenciamento de mudanças, organização de papeis e responsabilidades para alcançar sucesso constante.

Modelo de Qualidade CMM - Capability Maturity Model

O CMMi baseado na melhoria, compõe-se de cinco níveis de maturidade que demonstram o grau de maturidade da organização. Cada nível possui processos definidos, sendo cada um deles compreendido por áreas chaves de processo que trazem os objetivos a serem atingidos garantindo estabilização. Na **Figura 5** podem ser observados os cinco níveis pelos quais as organizações evoluem:

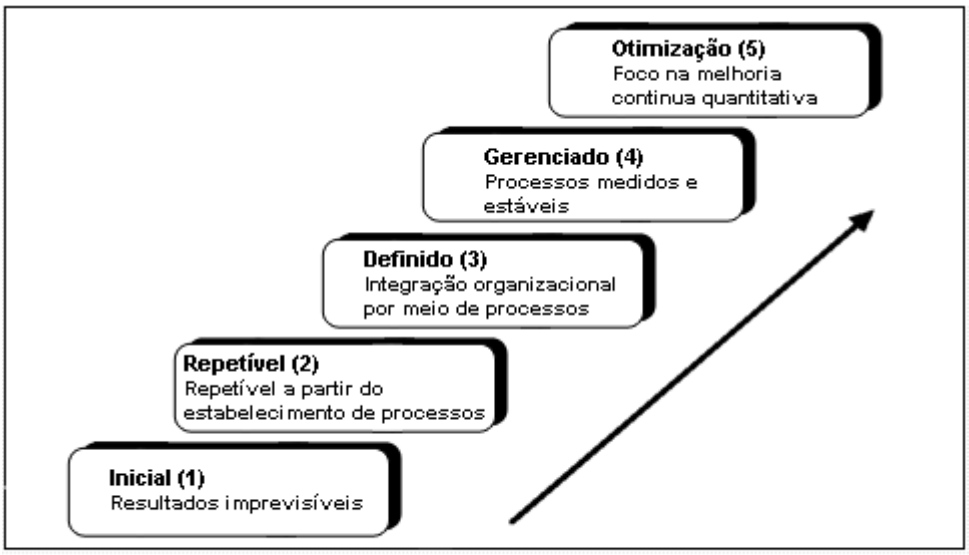

Figura 5. Os Níveis de Maturidade do CMMi

1. **Inicial** - É onde os processos são avaliados e definidos. Pode ser considerado imprevisível, pois o sucesso dessa fase apoia-se no esforço individual;
2. **Repetível** – Os níveis básicos dos processos são estabelecidos, como o monitoramento de custo, prazo e funcionalidade, para que estejam disciplinados, garantindo repetições em processos futuros;
3. **Definido** – Onde os processos são documentados e padronizados. As responsabilidades são estabelecidas a cada integrante dos projetos e suas atividades são detalhadas. Todos os projetos serão baseados nesse nível;
4. **Gerenciado** – Onde se inicia as previsões e o controle dos processos. São feitas medições e é observada a relação entre os processos e a qualidade. Nesse nível é possível prever resultados, pois é onde se coletam dados para o processo de qualidade;
5. **Otimização** – Melhora continua através de feedback dos processos acima e da condução de novas ideias.

Cada um destes níveis possibilita observar o amadurecimento da organização, resultando em um melhor produto.

Para alcançar as metas de cada nível, existem áreas chaves, que definem os procedimentos necessários a cada um. Estas são passos que devem ser seguidos para a conclusão do nível correspondente:

- Não possui áreas chave de processo;
- Gerenciar requisitos, planejar os processos do software, acompanhar o projeto do mesmo, gerenciá-lo, controlar a qualidade e gerenciar as configurações;
- Focalizar os processos da organização e defini-los. Aplicar treinamentos, gerenciar integralmente o software, aplicar a engenharia do produto, coordenar através de intergrupos e revisar detalhes para prevenir defeitos;
- Gerenciamento quantitativo dos processos e da qualidade do software;
- Prevenção de falhas, gerenciar mudanças nos processos, política de responsabilidade, recursos, estrutura e treinamentos. Planejamento, melhorias e correções. Revisão de detalhes para prevenção de defeitos.

Sendo cumpridos os níveis citados acima, a organização estará seguindo os princípios de qualidade do modelo CMMi, em consequência, terá garantia de que seus processos organizacionais estão seguros e confiáveis como descreve o processo.

ISO

A ISO está subdivida em normas técnicas ou Standards, cada uma destas se responsabiliza por uma determinada área. As principais para o assunto em destaque são a ISO/IEC 12207 e a ISO/IEC 15504.

A ISO/IEC 12207 foi publicada em agosto de 1995. Possui uma terminologia bem definida e serve de referência para a indústria do software. Ela fornece uma arquitetura para o ciclo de vida do mesmo, normas para aquisição, fornecimento, desenvolvimento operação e manutenção. Ela se divide em três processos que compõem o ciclo de vida:

- Processos primários: distribuição dos papéis que cada membro irá exercer, como adquirente, fornecedor, desenvolvedor, operador e quem o mantém;
- Processos de suporte: suportam outros processos como partes integrantes como documentação, configurações, gerenciamento de qualidade, verificação, revisão, auditoria e solução de problemas;
- Processos organizacionais: Estabelecidos e implementados em uma estrutura de processos associados. Composto por pessoas, nele ocorre o melhoramento contínuo.

A arquitetura da ISO/IEC 12207 baseia-se em dois princípios: modularidade e responsabilidade. Essa norma procura distinguir os papéis dos responsáveis pelo desenvolvimento do software, diferenciar em módulos o projeto e esclarecer a função de cada um.

A ISO/IEC 15504 realiza avaliações de processos de software com objetivo de melhorar processos e determinar a capacidade dos mesmos.

MPS.BR

O MPS.BR é um programa para melhoria da qualidade. É coordenado pela SOFTEX – Associação para promoção da excelência do software brasileiro. Conta com o apoio do MCT – Ministério da Ciência e Tecnologia, da FINEP – Financiadora de Estudos e Projetos e do BID – Banco Interamericano de Desenvolvimento.

Ele está descrito por documentos em formas de guias. A Guia Geral descreve de forma detalhada o Modelo de Referencia (MPS.BR) e fornece uma visão geral sobre as demais guias que apoiam a implementação dos diversos níveis do processo, facilitando a compreensão de todos.

A Guia de Avaliação descreve o processo e método de avaliação, tanto para avaliados como para avaliadores. A Guia de Aquisição serve como apoio para as instituições interessadas no método de como adquiri-lo e a Guia de Implementação descreve cada nível.

A base técnica para a construção e aprimoramento deste modelo de melhoria e avaliação de processo de software é composta pelas normas NBR ISO/IEC 12207 – Processo de Ciclo de Vida de Software, pelas emendas 1 e 2 da norma internacional ISO/IEC 12207 e pela ISO/IEC 15504 – Avaliação de Processo. Uma avaliação MPS.BR é realizada utilizando o processo e método de avaliação MA-MPS descritos no guia de avaliação. Uma avaliação MPS.BR verifica

a conformidade de uma organização/unidade organizacional aos processos do MR-MPS. O MPS.BR é definido em consonância com a norma internacional ISO/IEC 12207, adaptando-a às necessidades da comunidade de interesse. O MR-MPS foi definido em conformidade ao CMMI-DEV. Para definição e revisão do modelo de referência é feita uma ampla consulta à comunidade de implementadores e avaliadores MPS.BR. A elaboração final é responsabilidade da ETM.

A **Figura 6** ilustra a base técnica do modelo MPS.BR: a ISO e o CMMI.

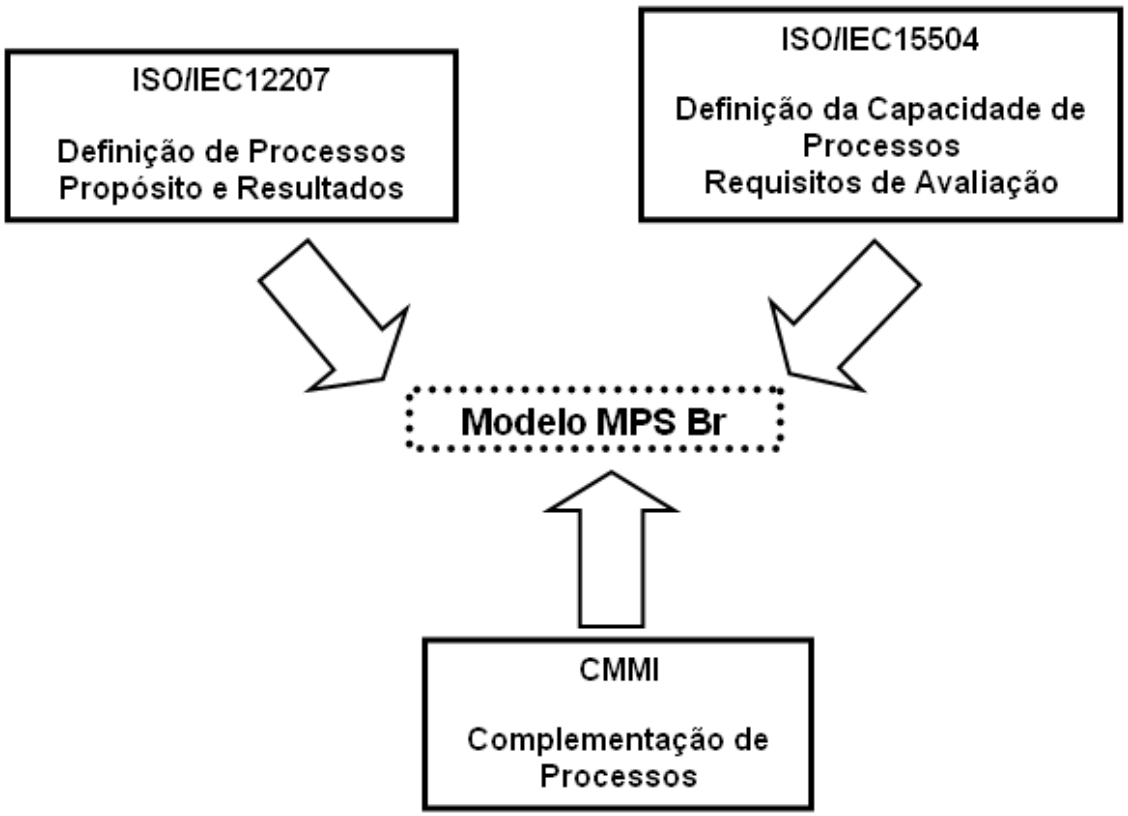

Figura 6. Base técnica do modelo MPS.BR

O MPS-BR define níveis de maturidade, estes são evolutivos tendo estágios para implementação de processos. O nível em que se encontra uma organização possibilita prever o desempenho futuro da empresa. O MPS-BR é definido em sete níveis de maturidade:

- **A** (Em otimização);
- **B** (Gerenciado Quantitativamente);
- **C** (Definido);
- **D** (Largamente Definido);
- **E** (Parcialmente Definido);
- **F** (Gerenciado) e
- **G** (Parcialmente Gerenciado).

A escala de maturidade se inicia no nível **G** e progride até o nível **A**. O progresso e o alcance de próximo nível são obtidos quando todos os propósitos pré-estabelecidos são atendidos e os resultados esperados alcançados.

Embora a avaliação por níveis baseie-se no CMMI, há diferenças, pois se tem o objetivo de possibilitar uma implementação e avaliação adequada a micros, pequenas e médias empresas,

possibilitando a realização de avaliações considerando mais níveis e também permitindo resultados em prazos menores.

Além dos modelos que as organizações podem estar adaptando no seu dia a dia, é indispensável que os produtos desenvolvidos passem pela fase de testes.

Teste de software

Diante de todos os processos, modelos e ferramentas que a engenharia do software disponibiliza para obter qualidade dos produtos desenvolvidos, ainda assim pode ocorrer incidência de erros. Com o objetivo de minimizar estes, as técnicas de teste têm se mostrado importantes.

O teste é uma das fases do processo de engenharia de software que tem por objetivo avaliar se o produto desenvolvido cumpre com todas as especificações declaradas, ou seja, verificar se o produto atende aos requisitos definidos na fase inicial do projeto.

Testar um software é executá-lo de maneira controlada, analisando as conformidades e funcionalidades do projeto, tendo por objetivo encontrar erros, falhas e bugs antes da entrega da versão final ao cliente.

Testar os produtos de software envolve basicamente quatro etapas: planejamento, projetos de casos de teste, execução e avaliação dos resultados. Estas atividades devem ser realizadas dentro do próprio processo de desenvolvimento.

O padrão IEEE número 610.12-1990 (IEEE Standards Board, 1990) diferencia os termos normalmente citados na fase de testes:

- Defeito (fault): passo, processo ou definição de dados incorretos;
- Engano (mistake): ação humana que produz um resultado incorreto;
- Erro (error): diferença entre o valor obtido e o valor esperado, ou seja, qualquer estado intermediário incorreto ou resultado inesperado na execução do programa constitui um erro;
- Falha (failure): produção de uma saída incorreta com relação à especificação.

A **Figura 7** apresenta a diferença entre Defeito, Erro e Falha. Os defeitos são causados por pessoas e fazem parte do universo físico. A presença destes pode ocasionar os erros que se encontram no universo da informação. E estes podem gerar as falhas encontradas no universo do usuário.

Figura 7. Defeito x Erro x Falha

De uma forma geral, os erros são classificados em:

- Erros computacionais: o erro provoca uma computação incorreta, mas o caminho executado (sequências de comandos) é igual ao caminho esperado;
- Erros de domínio: o caminho executado é diferente do caminho esperado.

A atividade de testes se divide em três fases: unidade, integração e de alto nível. Esta divisão permite que o testador se concentre em aspectos diferentes do software e em diferentes tipos de erros utilizando estratégias de seleção de dados diversificadas.

Tipos de teste de software

A cada dia vem surgindo novos tipos de testes que podem ser adotados pelas empresas, dentre eles estão:

- Teste de desempenho ou performance: verifica se o desempenho está consistente com os requisitos definidos empregando um volume de transações nas situações de máximo de acesso e concorrência para avaliar os resultados encontrados com os especificados;
- Teste de carga: avalia e mede os comportamentos de desempenho e sua capacidade funcional sob diferentes cargas de trabalho com grandes volumes de dados;
- Teste de stress: submete o sistema a uma carga desproporcional impedindo o uso dos recursos necessários para o processamento;
- Teste de código: verifica se o código está de acordo com as padronizações de linguagem, estruturação, endentação, nomes de arquivos, variáveis e funções com o objetivo de deixá-lo estruturado, legível e reusável;
- Teste de usabilidade: verifica a facilidade que o software possui de ser compreendido e manipulado pelo usuário, ou seja, se possui manuais, help on-line e sistemas eletrônicos;
- Teste de segurança e controle de acesso: verifica a segurança do usuário ao efetuar login no sistema. Os acessos aos dados disponíveis para somente quem possui permissão ter acesso aos dados restritos;
- Teste de integridade de dados: avalia a robustez quanto à resistência a falhas e a compatibilidade técnica em relação à linguagem, sintaxe e utilização de recursos;

- Teste de interface do usuário: verifica a interação do usuário com o sistema, assegurando navegação adequada e que os objetos contidos na interface funcionem corretamente;
- Teste de instalação e configuração: avalia a instalação sob diversas circunstancias e em diferentes plataformas de hardware e software, garantindo o devido funcionamento;
- Teste de unidade: verifica a menor unidade de projeto de software: módulos, métodos, classes e também trechos de código confusos. O objetivo dele é assegurar que cada unidade está funcionando de acordo com sua especificação funcional;
- Teste de integração: verifica se as unidades testadas individualmente (teste unitário) quando integradas executam corretamente. Os testes de integração dividem-se em dois tipos: incremental e não-incremental.

Na abordagem incremental o software é construído e testado em blocos. Isso facilita o isolamento e a correção de erros. Neste processo podem ser utilizadas duas estratégias: descendente (**top-down**) ou ascendente (**bottom-up**).

Na abordagem não incremental todos os módulos são combinados antecipadamente e o programa completo é testado.

- Teste de sistema: seu objetivo é assegurar que o software e os demais elementos que o compõem se combinam adequadamente e que a função desejada seja obtida;
- Teste de validação ou aceitação: é realizado com o propósito de avaliar a qualidade externa do produto e também na medida do possível a qualidade em uso;

Geralmente é realizado de duas maneiras: teste alfa e beta. O primeiro é realizado no ambiente de desenvolvimento com os usuários finais onde o testador registra os erros e problemas de uso. O segundo é realizado pelo usuário final em seu próprio ambiente, registrando os problemas encontrados e relatando-os ao desenvolvedor.

- Teste de regressão: objetiva garantir que o software permaneça funcional depois que mudanças no software sejam realizadas;
- Testes Back-to-back: o mesmo teste é realizado em versões diferentes e o resultado é comparado;
- Teste de recuperação: valida a capacidade e qualidade da recuperação do software após erros catastróficos ou falhas de hardware;
- Teste de alterações: rastrear alterações durante o processo de teste;
- Teste de recuperação de versões: verifica a capacidade de voltar a uma versão antiga do sistema;
- Teste de interoperabilidade: avalia as condições de integração com outros softwares e ambientes;
- Teste de sobrevivência, confiabilidade e disponibilidade: avalia a capacidade de permanecer operando quando algum elemento para de funcionar;
- Teste estático: avalia toda a documentação do projeto como modelos, requisitos entre outros;
- Teste embutido: avaliar integração entre hardware e software.

Técnicas de teste de software

As técnicas de teste fornecem diretrizes para projetar testes que exercitam a lógica interna dos componentes do software, assim como seus domínios de entrada e saída. Dentre as técnicas propostas, encontram-se as técnicas de testes: funcional, estrutural e baseada em erros.

A técnica de teste funcional ou caixa preta verifica o software como se fosse uma caixa da qual o conteúdo é desconhecido, sendo possível visualizar somente o lado externo, ou seja, os dados de entrada e saída. São utilizados os requisitos do programa para derivar os requisitos de teste que irão ser empregados sem se importar com detalhes de implementação.

Procura revelar erros como funções incorretas ou omitidas, erros de interface, erros nas estruturas de dados ou no acesso ao banco de dados externo, erros de desempenho, de inicialização e termino. O objetivo do teste funcional é encontrar oposições entre o comportamento atual do sistema e o que está descrito em suas especificações. Os critérios mais abordados nessa técnica são particionamento de equivalência e análise de valor limite.

A técnica de teste estrutural ou caixa branca leva em consideração os aspectos de implementação na escolha dos casos de teste. Os requisitos são estabelecidos com base na implementação do programa e a estrutura e detalhes do código são levados em consideração para a geração dos casos de teste. Trata-se de uma técnica do projeto de casos de teste que usa a estrutura de controle e o fluxo de dados para derivar os requisitos de teste. Os critérios baseiam-se em diferentes tipos de estruturas. Esses podem ser classificados em: baseados na complexibilidade, no fluxo de controle e em fluxo de dados. As técnicas baseadas em erros estabelecem os requisitos de teste explorando os erros típicos e comuns cometidos durante o desenvolvimento. As técnicas de teste são utilizadas de forma complementar, pois detectam categorias de erros distintas para que cada uma seja bem explorada e leve a um processo de boa qualidade e baixo custo.

Critérios de teste de software

Apenas a utilização de técnicas de teste não seria viável, por isso juntamente a elas são utilizados critérios de teste. Estes auxiliam o testador, fornecem um método de avaliação de conjuntos de casos de teste e uma base para seleção dos casos de teste.

Os critérios podem ser utilizados como:

- Critério de cobertura: permite a identificação de partes do programa que devem ser executados para garantir a qualidade e indicar quanto o mesmo foi suficientemente testado;
- Critério de adequação: a partir de um conjunto de casos de teste qualquer, ele é utilizado para verificar se satisfaz os requisitos estabelecidos pelo critério. Ou seja, este critério avalia se os casos de teste definidos são suficientes ou não para avaliação de um produto ou uma função;
- Critério de geração: quando o critério é utilizado para gerar um conjunto qualquer de casos de teste adequado para um produto ou função. Define as regras ou diretrizes para geração dos casos de um produto que esteja de acordo com o critério de adequação.

Metodologias de teste de software

Para atingir o objetivo do trabalho proposto, foi realizada uma pesquisa referente às metodologias existentes. Através da pesquisa realizada foram selecionados materiais que tratam de assuntos relacionados à qualidade de software, engenharia, testes e metodologias de teste. Nestes, foram analisadas e destacadas as metodologias e modelos de teste mais utilizados pelas empresas no período de 2007 até hoje.

Abaixo serão apresentados: Modelo IDEAL, Metodologia do CenPRA, Modelo “V” e Metodologia 3P x 3E. Serão descritas as principais características de cada metodologia, possibilitando selecionar entre elas a mais viável para aplicação na empresa AmSoft

Modelo IDEAL

O modelo IDEAL proposto pelo SEI (Software Engineering Institute) da Carnegie Mellon University dos EUA tem como objetivo fornecer uma guia para direcionar iniciativas de programas de melhoria de software através de um longo e integrado plano para iniciação e gerenciamento. As atividades propostas estão em um alto nível de abstração. O modelo é dividido em cinco fases (a **Figura 8** ilustra as fases do modelo IDEAL e as principais atividades de cada fase):

- Inicialização (Initiating): Preparação e criação de um plano de melhoria do processo de teste. Nesta etapa serão definidas as metas a serem alcançadas, a infraestrutura a ser utilizada, as responsabilidades e os recursos (humanos e físicos);
- Diagnóstico (Diagnosing): Levantamento do estado atual da empresa e definição do estado desejado. Serão utilizados os resultados, recomendações dos validadores e outras iniciativas de melhorias para o plano de ação;
- Estabelecimento (Establishing): Planejamento detalhado de como alcançar as melhorias desejadas. A versão inicial do plano de melhoria do processo de teste será elaborada de acordo com a visão da empresa, plano estratégico de negócios, lições apreendidas de experiências passadas;
- Ação (Acting): Execução do planejamento. Nessa fase soluções endereçadas para os pontos de melhoria diagnosticados são criadas, testadas e implantadas na empresa. Planos são desenvolvidos para execução de pilotos a fim de testar e avaliar os processos novos alterados. Com o sucesso destes, a implantação, institucionalização e planos para extensão são desenvolvidos e executados;
- Aprendizagem (Learning): Aprendizado adquirido durante o processo de melhoria. O objetivo dessa fase é fazer com que o próximo passo atrás do modelo IDEAL seja mais efetivo. Neste momento, soluções já foram desenvolvidas, lições já foram aprendidas, métricas coletadas e objetivos alcançados. Os artefatos são adicionados à base de dados do processo que irá se tornar uma fonte de informação para as pessoas envolvidas no próximo passo do modelo.

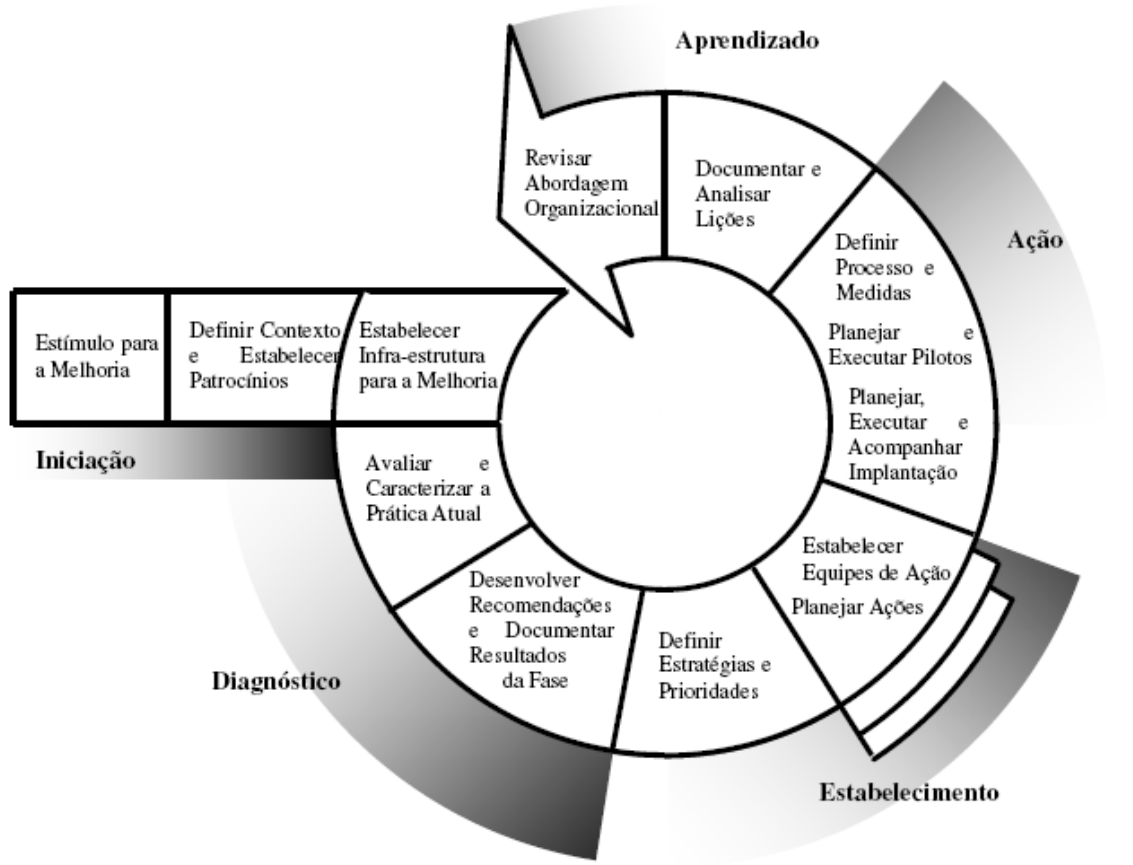

Figura 8. Fases do modelo IDEAL

Metodologia do CenPRA

A metodologia desenvolvida pelo CenPRA (Centro de Pesquisas Renato Archer) é fundamentada nos modelos sugeridos pela norma IEEE 829, que descreve os documentos que devem ser gerados na atividade de gerência dos testes de software.

A metodologia utiliza-se de técnicas, procedimentos e ferramentas para melhorar o processo de teste capacitando às empresas produtoras a desenvolver produtos com maior qualidade.

Ela foi desenvolvida de maneira que as empresas pudessem aplicá-la a vários tipos de sistema, como sistemas de informação e sistemas específicos. Também atende diferentes tipos de organização desde a microempresa até as de grande porte, já que cada uma pode selecionar a melhor maneira como aplicá-la.

O processo de teste não contempla automação do teste, esta decisão foi tomada para manter a descrição inicial do processo simples. Este fato revela que um processo só deve ser suportado por ferramentas quando estiver convenientemente definido e consistentemente adotado.

O processo de automação de teste tem maior probabilidade de ser bem sucedido para organizações que possuam uma equipe de teste bem definida e com um processo padrão de documentação seguido.

A metodologia do CenPRA baseia-se em três componentes:

- Treinamento: capacitação em conceitos básicos sobre teste, técnicas de teste, documentação e processo de teste;
- Processo de teste: realização das atividades de planejamento, projeto, execução e acompanhamento dos testes;
- Suporte para geração de documentos: aplicação de técnica para criação dos documentos baseado na norma IEEE.

O processo de teste proposto na metodologia está baseado em alguns pressupostos básicos:

- Os testes de sistema e aceitação são projetados e executados sob a responsabilidade da equipe de teste;
- Os testes de sistema, e eventualmente também o de aceitação, são realizados de forma iterativa, havendo, antes do início de cada ciclo de teste, uma avaliação rápida do produto.

A geração dos documentos que fazem parte do componente “Suporte para Geração de Documentos” está baseada na norma IEEE 829-1998. São eles (a **Figura 9** ilustra o relacionamento entre os documentos de teste):

- Plano de Teste: apresenta o planejamento incluindo a abrangência, abordagem, recursos e cronograma. Identifica os itens e as funcionalidades a serem testados as tarefas e riscos associados;
- Especificação de projeto de teste:
    - Projeto de teste: identifica as funcionalidades e características a serem testadas e também os casos e procedimento de teste e apresenta os critérios de aprovação;
    - Casos de teste: define os casos de teste, incluindo dados de entrada, resultados esperados ações e condições gerais para a execução do teste;
    - Procedimentos de teste: especifica os passos para executar um conjunto de casos de teste.
    - Relatórios de Teste
    - Relatório de Passagem de Itens de Teste: identifica os itens encaminhados para teste no caso de equipes distintas serem responsáveis pelas tarefas de desenvolvimento e de teste;
    - Relatório de Log de Teste: apresenta registros cronológicos dos detalhes relevantes relacionados com a execução dos testes;
    - Relatório de Incidentes de Teste: documenta qualquer evento que ocorra durante a atividade de teste e que requeira análise posterior.
    - Relatório de Resumo de Teste: apresenta de forma resumida os resultados das atividades de teste associadas com uma ou mais especificações de projeto de teste e provê avaliações baseadas nesses resultados.

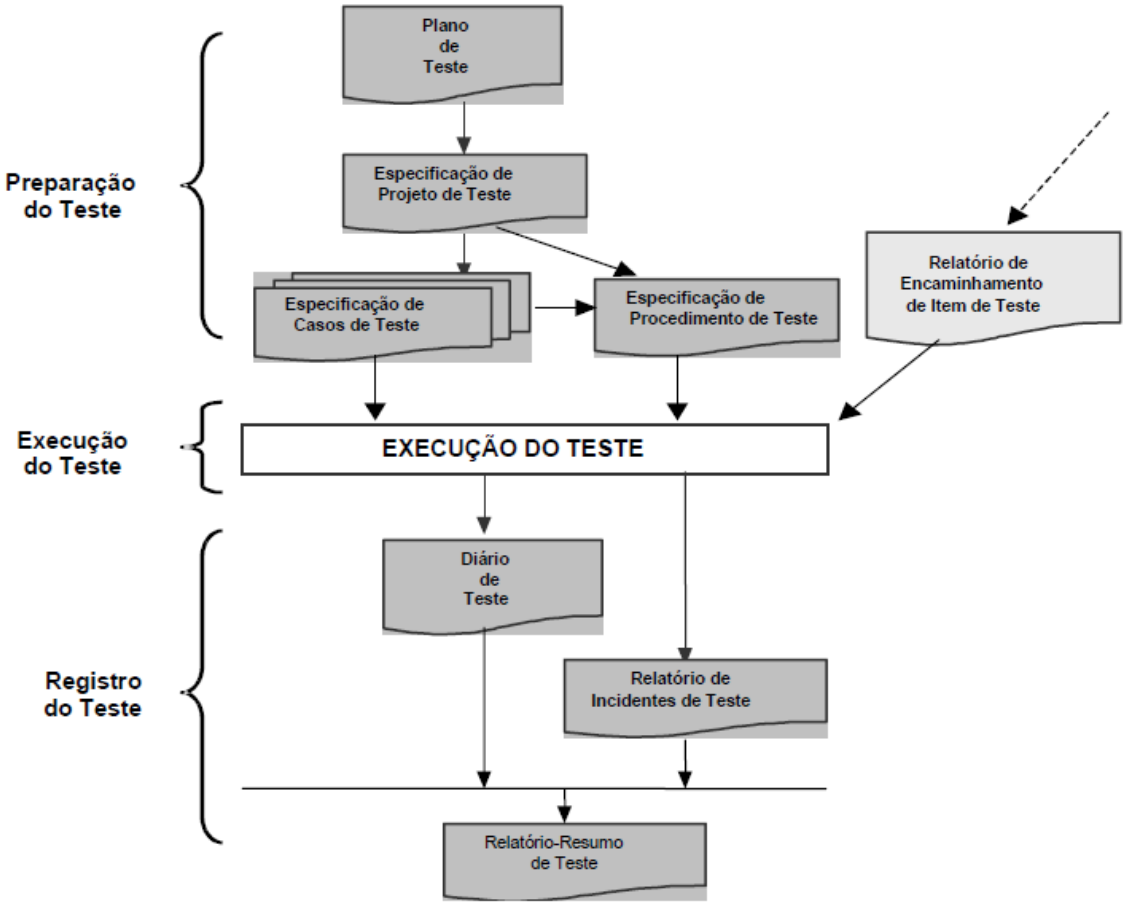

Figura 9. Relacionamento entre os documentos de teste Modelo "V"

No Modelo "V" os testes são iniciados juntamente com o desenvolvimento de software, ou seja, enquanto a equipe de desenvolvimento dá início ao projeto do sistema a equipe de teste inicia o planejamento dos mesmos. As duas equipes partem do mesmo ponto com as mesmas informações.

No modelo "V" de testes predominam do início ao fim os procedimentos de "fazer" e "conferir". A equipe que "faz" trabalha na implementação do sistema e a equipe que "confere" executa os testes com o objetivo de minimizar erros.

A **Figura 10** apresenta o modelo "V" de teste de software. Do lado esquerdo é representado o desenvolvimento do software e seus respectivos níveis. Do lado direto as fases do teste.

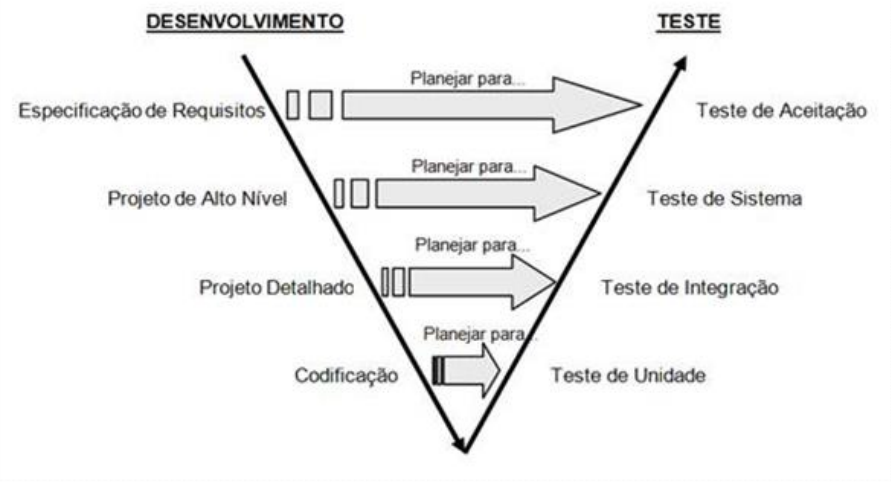

Figura 10. Modelo "V": paralelismo entre as atividades de desenvolvimento e teste de software

Os passos do processo de teste de software no modelo "V" são:

- Acesso ao plano de desenvolvimento: nessa etapa os testadores verificam se o plano de desenvolvimento está completo e correto, assim podem estimar os recursos necessários para efetuar os testes. Com estas informações terão condições de desenvolver o plano de teste;
- Desenvolvimento do plano de teste: preparação do plano de teste de acordo com os requisitos levantados no plano de desenvolvimento;
- Inspeção ou teste de requisitos de software: avaliação dos requisitos utilizando a técnica de verificação;
- Inspeção ou teste de desenho do software: verifica se o projeto atinge os objetivos dos requisitos através da técnica de verificação;
- Inspeção ou teste de construção do software: determinação do tipo e extensão dos testes necessários de acordo com o método escolhido para construir o software;
- Execução dos testes: testa o código em estado dinâmico, utilizando-se das especificações, ferramentas e métodos especificados no plano de teste;
- Teste de aceitação: avaliação da aplicabilidade e usabilidade do sistema pelos usuários;
- Informação dos resultados de teste: pode ser verbal ou por escrito repassando aos setores envolvidos os problemas;
- Teste de implementação de software: verifica a interoperabilidade com o sistema operacional e outros softwares e também com os procedimentos operacionais;
- Teste de mudanças no software: verifica as mudanças durante o processo de implementação e após a ela.
- Avaliação da eficácia dos testes: avalia a eficácia e melhorias no processo.

.

Metodologia 3P x 3E

A metodologia de teste 3P x 3E foi confeccionada para servir de modelo básico para elaboração e monitoração do processo de testes. Os 3P´s representam Procedimentos, Planejamento e Preparação e os 3E´s Especificação, Execução e Entrega.

O modelo 3P x 3E está dividido em três etapas: procedimentos, planejamento e preparação. Cada etapa é composta de atividades, produtos e documentos e estas possuem um percentual estimado dentro do ciclo de vida do processo de teste:

- Procedimentos iniciais: menor etapa do modelo, elaboração do processo de teste e assinatura de um acordo de nível de serviço executado;
- Planejamento: planejar atividades de estratégia de testes, planos de testes e revisões;
- Preparação: adequar o projeto de testes a gerência de configurações ou controle de mudanças;
- Especificação: é realizada a elaboração dos casos de teste e/ou roteiros;
- Execução: são preparados os dados de teste, os testes são executados e ocorre a solução das ocorrências. Nesta fase também ocorre o monitoramento da execução dos testes e elaboração do relatório final;
- Entrega: é a finalização do modelo onde são avaliados e arquivados os artefatos e dados etapas anteriores.

A **Figura 11** nos mostra o ciclo de vida do processo de teste 3P x 3E.

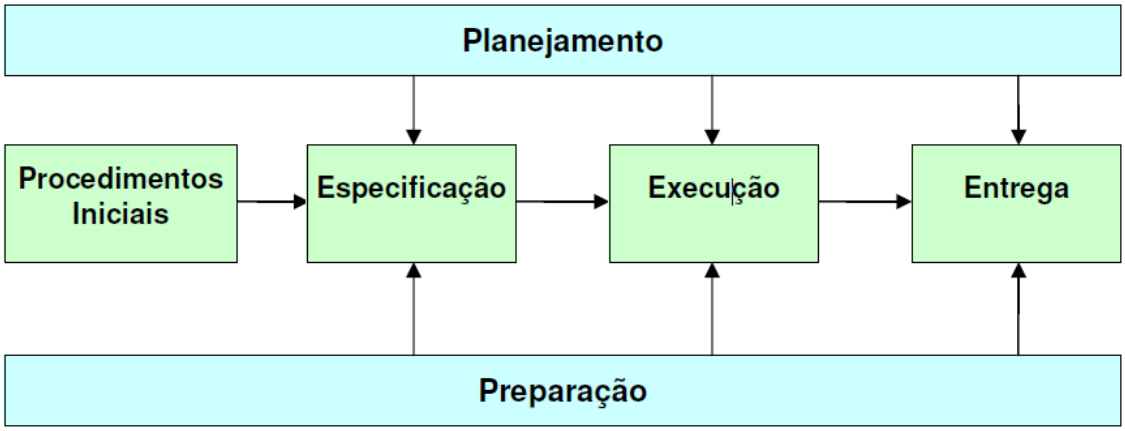

Figura 11. Ciclo de vida do processo de teste

A sequência de etapas da execução dos testes é a seguinte:

1. Procedimentos iniciais;
2. Planejamento;
3. Preparação;
4. Especificação;
5. Execução;
6. Entrega.

Os principais documentos gerados nesta metodologia são: Guia Operacional de Testes, estratégias de testes, planos de testes, casos de testes e roteiros de testes.

No modelo 3P x 3E também é recomendado o início dos testes paralelo ao desenvolvimento. O início dos testes próximo ao início do desenvolvimento tem por objetivo a redução de custos. O custo de correção de defeitos cresce exponencialmente à medida que o projeto de desenvolvimento avança. A **Tabela 2** apresenta as etapas e sub-etapas do modelo 3P x 3E.

Etapa	Sub-Etapa	Insumo	Produto
Procedimentos iniciais	1) Elaborar o guia operacional do teste	· Requisitos do negocio · Modelos de dados · Diagramas de fluxo de dados · Diagramas de contexto · Outros documentos de desenvolvimento	Guia operacional de testes
Planejamento	2.1) Estabelecer estratégia de Testes	· Requisitos do Negócio · Modelos de dados · Diagrama de fluxo de dados. · Outros documentos de Desenvolvimento · Guia Operacional de Teste	Estratégia de Teste Análise de Riscos do Projeto de Teste
	2.2) Estabelecer Plano de Teste	· Estratégia de Teste · Análise de Riscos do Projeto de Teste · Necessidades de dados de testes · Planejamento do sistema que está desenvolvendo	Plano de testes nova versão Análise de Riscos do Projeto de Testes
	2.3) Revisar Estratégia de Testes	· Requisitos de negócio do sistema · Estratégia de Teste	Estratégia de Testes revisada
	2.4) Revisar Plano de Testes	· Estratégia de Testes · Plano de Testes	Plano de Testes revisado

Preparação	3.1) Adequar o projeto de testes à Gerência de Configuração e/ou de controle de mudanças	· Arquitetura do ambiente de desenvolvimento · Arquitetura do ambiente de produção · Ferramentas e procedimentos de Gerência de Configuração e de mudança	Registro e controle das diversas versões do produto: funcional, desenvolvimento, produto e operacional
	3.2) Disponibilizar infraestrutura e ferramentas de teste	· Estratégia de testes · Arquitetura básica do ambiente de desenvolvimento · Arquitetura básica do ambiente de produção · Ferramentas de teste	Infraestrutura e ferramentas de teste disponíveis para a equipe de teste
	3.3) Disponibilizar pessoal	· Estratégia de Testes · Plano de Testes · Ferramentas de testes · Definição do ambiente de teste	Equipe de testes definida e capacitada

Especificação	4.1) Elaborar casos de teste	· Estratégia de testes · Plano de Testes · Documentação técnica do sistema · Necessidades de dados de teste · Posição quanto aos testes já realizados	Casos de Testes "Scripts"de testes (se usar ferramentas) Especificação das necessidades de dados de testes
	4.2) Elaborar Roteiros de Teste	· Casos de Testes · Planos de Teste · Fluxo de execução dos programas previsto pela equipe de desenvolvimento	Roteiros de Testes

Execução	5.1) Preparar dados de testes	· Casos de Testes · "Scripts"de Testes · Roteiros de Testes · Documentação do Sistema · Especificação das necessidades de dados de testes · Processos de criação de bases e/ou arquivos de teste	Bases/Arquivos de teste disponíveis
	5.2) Executar testes	· Roteiros de Testes · Casos de Teste · "Scripts"de Testes · Resultados Esperados	Resultados dos testes Relatórios de defeitos encontrados Ajustes no material de testes
	5.3) Solucionar ocorrência de testes	· Relatórios de defeitos com status a resolver · Resultados dos testes	Relatórios de defeitos encontrados com status resolvido ou a avaliar
	5.4) Acompanhar a execução dos Casos de Testes	· Relatórios de defeitos (resumo) · Resultados dos testes (resumo) · Estratégia de teste · Plano de Testes · Casos de Testes · Roteiros de Testes	Análise do andamento dos Casos de Testes
	5.5) Elaborar Relatório Final	· Análise dos resultados de teste · Estratégia de Testes · Resultados de testes · Relatórios de defeitos - resumo · Plano de testes	Relatório final dos testes
Entrega	6.1) Avaliação e Arquivamento da documentação	· Documentos de testes	Relatórios de não conformidade Relatório final de testes Documentação arquivada

Tabela 2. Etapas e sub etapas do modelo 3P x 3E

Critérios de avaliação e seleção

Os critérios de avaliação e seleção de uma metodologia a ser aplicada de acordo com as necessidades atuais da empresa são: custos, compreensão e facilidade de aplicação e tempo de resultado.

A análise dos custos é extremamente importante para a seleção da metodologia. Por se tratar de uma empresa de pequeno porte, os recursos disponibilizados são limitados. A compreensão

e facilidade de aplicação simplificam o trabalho, já que a equipe atual de colaboradores que vão trabalhar na implantação da metodologia é pequena. O tempo de resultado influencia na decisão devido às limitações que a empresa tem por ser de pequeno porte.

Em constante busca por crescimento, o objetivo é ter o resultado o mais rápido possível, para disponibilizar produtos melhores que lhe darão lugar de destaque no mercado em que atua. Diante destes fatores a busca é por uma metodologia que traga resultados positivos em prazo considerável sem fugir da realidade atual da empresa.