Verificação, Validação e Teste

A T2Ti

Apresentação

A T2Ti nasce do sonho de três colegas que trabalhavam no maior banco da américa latina.

Tudo começa em 2007 com o lançamento do curso Java Starter. Logo depois veio o Siscom Java Desktop seguido de outros treinamentos.

Desde então a Equipe T2Ti se esforça para produzir material de qualidade que possa formar profissionais para o mercado, ensinando como desenvolver sistemas de pequeno, médio e grande porte.

Um dos maiores sucessos da Equipe T2Ti foi o Projeto T2Ti ERP que reuniu milhares de profissionais num treinamento dinâmico onde o participante aprendia na prática como desenvolver um ERP desde o levantamento de requisitos. Foi através desse treinamento que centenas de desenvolvedores iniciaram seu negócio próprio e/ou entraram no mercado de trabalho.

Em 2010 a T2Ti lança sua primeira aplicação para produção, o Controle Financeiro Pessoal. O sucesso foi tanto que saiu até em matéria no site Exame, ficando entre os 10 aplicativos mais baixados da semana.

Começa então a era de desenvolvimento de sistemas para alguns clientes exclusivos, pois o foco ainda era em desenvolvimento de treinamentos. A T2Ti desenvolve sistemas para o mercado nacional e internacional.

Atualmente a T2Ti se concentra nas duas vertentes: desenvolver sistemas e produzir treinamentos.

_____

Sumário

Verificação, Validação e Teste

Teste
Introdução; Definições; Conceitos; Visão; BUG; Defeito; Teste e Qualidade;

Verificação de Software
Histórico; Teste de Verificação;

Validação do Software
Teste de Validação; Caixa Preta; Caixa Branca;

Dimensões do Teste
Introdução; Dimensão 1 – Estado do Teste;
Dimensão 2 – Técnica do Teste; Dimensão 3 – Metas do Teste; Dimensão 4 – Ambiente do Teste;

Processo de Teste
Introdução; Modelo "V"; Profissional de Teste; Considerações;

Teste

Introdução

Nas décadas de 1960 e 1970, os desenvolvedores dedicavam a maior parcela dos seus esforços nas atividades de codificação e nos testes unitários. Estima-se que 80% desse esforço eram para esse tipo de atividade. Uma parcela menor era dedicada à integração dos programas e nos testes dos sistemas. As atividades de teste eram consideradas um mal necessário para provar aos usuários que os produtos funcionavam e não eram tratados como um processo formal alinhado com as atividades do processo de desenvolvimento de sistemas.

A partir dos anos 1980, durante o processo de desenvolvimento, passou a ser dada maior importância a analise dos requisitos, ao desenho funcional e técnico dos novos sistemas.

Nós últimos anos, com a utilização da Internet para a realização de negócios, houve uma mudança significativa na abrangência e complexidade das aplicações, onde fatores, tais como segurança e performance passam a ser relevante, tornando a atividade de testar cada vez mais especializada.

Teste

Definições

Existem várias definições para Teste, segundo alguns autores:

Testar é verificar se o software está fazendo o que deveria fazer, de acordo com os seus requisitos, e não está fazendo o que não deveria fazer (Rios e Moreira – 2002).

Testar é o processo de executar um programa ou sistema com a intenção de encontrar defeitos (teste negativo) (Glen Myers - 1979).

Testar é qualquer atividade que a partir da avaliação de um atributo ou capacidade de um programa ou sistema, seja possível determinar se ele alcança os resultados desejados (Bill Hetzel – 1998).

Muitas outras definições poderiam ser ainda citadas, mas em essência, teste de software é o processo que visa executar o software de forma controlada, com o objetivo de avaliar o seu comportamento, baseado no que foi especificado. A execução dos testes é considerada um tipo de validação.

Teste

Conceitos

Muitas empresas não se preocupam com testes colocando desenvolvedores menos qualificados para os realizar, pois para elas qualquer um pode testar, entretanto quando desenvolvedores os testam querem provar que "algo funciona", já o analista de teste procura provar que algo não funciona.

O fato é que é mais fácil provar que "algo funciona" do que provar que "algo não funciona".

Quando falamos que o software foi testado, entendemos que está livre de erros. Quem pensar assim estará se enganando, pois erros sempre existiram. O que se faz é diminuir o risco de ocorrer um erro.

Testes servem para diminuir o risco.

Consideremos então alguns conceitos básicos relacionados ao universo dos testes de software.

Teste

Conceitos

Base de Teste (Test Basis)

Todos os documentos a partir dos quais os requisitos de um determinado componente ou sistema podem ser inferidos. Documentação na qual os casos de testes estão baseados.

Se um documento pode ser alterado somente por meio de procedimento formal, então a base de teste passa a se chamar base de teste congelada.

Caso de Teste (Test Case)

Conjunto de valores de entrada/inputs, precondições de execução, resultados esperados e pós-condições de execução desenvolvidas para um determinado objetivo ou condição de teste, tais como para exercitar o caminho de um determinado programa ou verificar o atendimento a um requisito especifico.

Teste

Conceitos

Execução de Teste (Test Execution)

Processo de executar um teste em um componente ou sistema sendo testado e que produz resultados reais.

Fase de Teste (Test Phase)

Conjunto distinto de atividades de teste coletadas em uma fase gerenciável do projeto.

Gerente de Teste (Test Manager)

Pessoa responsável pelo gerenciamento do projeto, pelas atividades e recursos de teste e por avaliar o objeto de teste. É o indivíduo que dirige, controla, administra, planeja e regula a avaliação de um objeto de teste.

Objetivo de Teste (Test Objective)

Razão ou finalidade por trás da modelagem e da execução de um teste.

Teste

Conceitos

Nível de Teste (Test Level)

Grupo de atividades de teste organizadas e gerenciadas conjuntamente. Um nível de teste está ligado às responsabilidades do projeto. Podemos citar como exemplos: teste de componente, teste de integração, teste de sistema e teste de aceitação.

Testador (Tester)

Profissional habilitado e envolvido no teste de um componente ou sistema.

Política de Teste (Test Policy)

Documento de alto nível que descreve os princípios, a abordagem e os principais objetivos da organização de um teste.

Resultado de Teste (Test Result)

Consequência da execução de um teste. Inclui saídas/outputs para telas, alterações de dados, relatórios e comunicações enviadas. Ver também resultado real e resultado esperado.

Teste

Conceitos

Software (Software)

Programas e procedimentos de computação, e possível documentação associada e dados pertinentes à operação de um sistema de computador.

Testware (Testware)

Artefatos produzidos durante o processo de teste e requeridos para planejar, projetar e executar testes, entre eles documentação, roteiros, entradas/inputs, resultados esperados, procedimentos de preparação e de esclarecimento, arquivos, bancos de dados, ambiente e qualquer software adicional ou utilitários utilizados no teste.

Teste

Visão

Algumas pessoas têm uma visão errada do profissional de teste, pois pensam que ele está ali para ser "inimigo dos desenvolvedores e analistas de requisitos" e para apontar quem não está fazendo seu trabalho direito.

Visão do Analista de Sistemas

1. Provar que as coisas estão funcionando.
2. Idealizar um conjunto de cenários favoráveis à utilização de um software ou um documento qualquer. Apresentar os cenários positivos.

Visão do Analista de Testes

1. Provar a não adequação de algo.
2. Estender nossa abstração e identificar um maior volume de cenários positivos.
3. Idealizar um conjunto de cenários desfavoráveis à utilização de um software ou um documento qualquer. Apresentar os cenários negativos.

Teste

BUG

O termo bug foi associado a interferências e mau funcionamento bem antes que existissem os computadores modernos, sendo Thomas Edison um dos primeiros a usar este significado.

Mas foi uma mulher, Grace Murray Hopper, quem em 1945 documentou o primeiro bug da informática.

A palavra traduzida literalmente do inglês como inseto, adquire outro significado quando falamos de informática.

Esta outra acepção se refere a elementos e circunstâncias no software ou hardware, involuntários e indesejados, que provocam um mau funcionamento.

Ao longo dos anos este termo se popularizou e hoje em dia utilizamos frequentemente para referir-se aos erros nos programas de computador.

Teste

BUG

Grace Brewster Murray Hopper (1906-1992), doutora em matemática pela universidade de Yale, passou à história por ser uma inovadora programadora durante as primeiras gerações de computadores.

Em 1943, durante a segunda guerra mundial, decidiu incorporar-se à marinha americana e foi enviada para o laboratório de cálculo Howard Aiken na Universidade de Harvard, onde trabalhou como programadora no Mark I.

Em 9 de setembro de 1945 o grupo de trabalho de Aiken e Grace se encontrava na sala do Mark II tentando averiguar porquê o computador não funcionava adequadamente. Depois de um exame minucioso conseguiram detectar o problema, uma mariposa entre os contatos de uns dos relés do Mark II. Mais tarde, Grace registraria o incidente no caderno de registros, colou a borboleta que causou o problema e anotou embaixo a frase "First atual case of bug being found". A partir de então, cada vez que algum computador dava problemas eles diziam que tinha bugs.

BUG – O Primeiro BUG Documentado

Photo # NH 96566-KN First Computer "Bug", 1945

9/9

0800 Arctan started

1000 " stopped - arctan ✓

13" u/c (032) MP - MC

(033) PRO 2

correct

Relays 6-2 in 033 failed special speed test

In relay

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Multy Adder Test.

1545 Relay #70 Panel F

(moth) in relay.

First actual case of bug being found.

1700 closed down.

Teste

BUG

"Bug" é falha, ou erro em um programa computacional, que o impede de se comportar como pretendido. A maioria dos erros é causada por compiladores produzindo um código inválido. Um programa no qual contenha vários erros e/ou erros que interferem seriamente na sua funcionalidade, é denominado de buggy. Os erros podem resultar em inúmeros efeitos, com níveis variando da inconveniência ao usuário do programa. Erros mais graves podem fazer com que o programa deixe de funcionar ou congele e conduza a negação de serviço.

Em pesquisas realizadas comprovou-se que a maioria dos erros encontrados é devido a requisitos mal elaborados. Se a funcionalidade desejada não é especificada ou comunicada de forma correta, seria como especificar algo incerto ou até incorreto.

Para testarmos devemos planejar os testes com precisão. O planejamento no teste é fundamental. Um "bug" na aplicação muitas vezes não é da aplicação em si, mas do processo de testes.

Teste

Defeito

Segundo Molinari, o defeito é uma não-conformidade em relação ao que o software se propõe a fazer, que diz que faz e não faz.

O defeito está associado a vários fatores como, por exemplo: o processo de software, a execução das atividades, o ambiente de trabalho, o hardware, o software e vários outros fatores podem auxiliar para que o defeito ocorra.

Os defeitos podem ser encontrados em toda a fase do processo de desenvolvimento do software, causando retrabalho, aumentando os custos do projeto, dilatando os prazos de entrega do software, diminuindo a produtividade e aumentando a insatisfação do cliente.

Devemos entender que testar nas fases iniciais é mais barato do que nas fases mais tardias do processo de desenvolvimento

Defeito – Tipos

Defeitos da funcionalidade (interface): Ocorrem quando o software executa uma ação não especificada nos seus requisitos ou quando o requisito especifica uma coisa e o software faz outra.

Defeitos de usabilidade (interface): São defeitos decorrentes da dificuldade para se navegar em uma aplicação.

Defeitos de desempenho (interface): O programa não atende com rapidez necessária às solicitações do usuário, especialmente no caso dos sistemas interativos. Existem estudos que indicam um tempo máximo de 3 segundos como o tempo máximo que o usuário suporta sem desviar a sua atenção para outra coisa.

Defeitos de saída (interface): Os resultados apresentados pelo software não estão conforme definido pelas especificações fornecidas pelo usuário ou foram mal projetadas dificultando a sua análise.

Defeito – Tipos

Alguns existem porque o sistema não está preparado para lidar com possíveis defeitos!

Prevenção dos defeitos: O programa não se protege das entradas de dados não previstas que posteriormente são processadas de forma inadequada. O software pode, por exemplo, aceitar valores em branco em campos numéricos.

Recuperação dos defeitos: Mesmo detectando o defeito, o programa falha porque não trata adequadamente a situação.

Detecção dos defeitos: O programa não trata (ou trata inadequadamente) as indicações de defeitos resultados das suas operações, tais como: overflow, flags de defeitos, etc. Nestes casos deveria ser dado um aviso da ocorrência do problema.

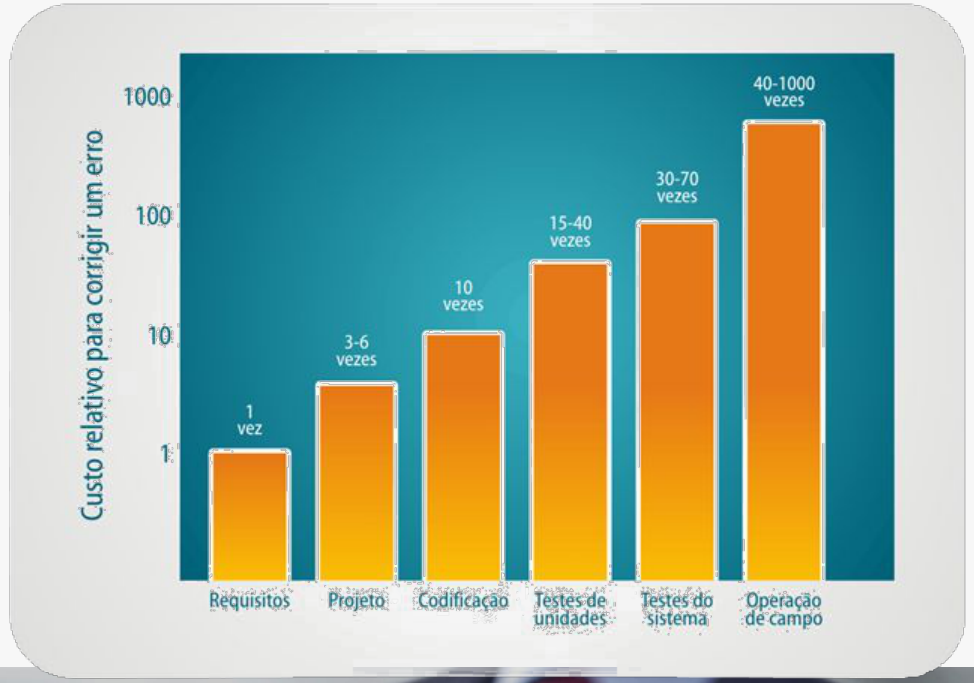
Defeitos de limites: O programa não consegue tratar ou trata inadequadamente valores extremos (o maior, o menor, o primeiro, o último, etc) ou fora dos limites estabelecidos.

Teste

Defeito – Custo

Todo e qualquer erro custa dinheiro e perdemos tempo para reconstruir algo defeituoso, fora os danos provocados pelo erro, o tempo que gastamos para identificar, corrigir, testar e implantar a correção. Quanto mais tarde descobrimos os erros, mais caros eles se tornam.

Myers aplica a chamada "regra 10" aos custos da correção dos erros. Significa que, quando um erro não é identificado, os custos de sua correção multiplicam-se por 10 para cada fase em que o erro migra.



Teste e Qualidade

Com a ajuda do teste é possível medir a qualidade do software em termos de defeitos encontrados, por ambas as características de requisitos, funcionais ou não funcionais.

O teste pode dar a confiança na qualidade do software, se ele encontrar pouco ou nenhum defeito. A execução de um teste planejado adequadamente reduz em grande medida os riscos de um sistema. Quando os testes encontram defeitos, a qualidade do sistema aumenta quando estes são corrigidos.

Lições devem ser aprendidas de projetos anteriores. Através do entendimento das causas-raiz dos defeitos encontrados em outros projetos, os processos podem ser aprimorados de modo a prevenir com que aqueles erros se repitam, e consequentemente, melhorar a qualidade dos sistemas futuros.

Testes devem ser integrados como uma das atividades da área de qualidade.

Verificação de Software

Introdução

O processo de desenvolvimento de software pressupõe dois momentos bem definidos. O primeiro é a fase de coleta de informações de negócios e o planejamento da arquitetura do software. Nesse momento, não existem componentes tecnológicos, mas sim documentos que especificam o comportamento a ser assimilado pelo software a ser construído. É nesse primeiro momento que uma forma básica de testes destaca-se: os testes de verificação, também conhecido como testes estáticos.

A segunda fase do desenvolvimento do software caracteriza-se pela existência de um componente computacional. Devemos aplicar um conjunto de testes que avaliam a qualidade do produto desenvolvido em relação aos requisitos identificados nas fases anteriores – é neste momento que a outra forma básica de teste aparece: os testes de validação. A validação consiste em identificar defeitos e possíveis problemas de um componente pronto, enquanto a verificação busca avaliar se a construção do componente segue os requisitos predefinidos.

Verificação de Software

Teste de Verificação

Os testes de verificação podem ser entendidos como um processo de auditoria de atividades e avaliação de documentos gerados durante todas as fases do processo de engenharia de software. As verificações devem ser aplicadas a todos os produtos que são produzidos em cada etapa do processo.

A missão dos testes de verificação é avaliar se toda a documentação gerada e todas as atividades que estão sendo desempenhadas são conduzidas adequadamente.

A principal característica dos testes de verificação é o fato de não envolver o processamento de softwares. Os testes de verificação avaliam:

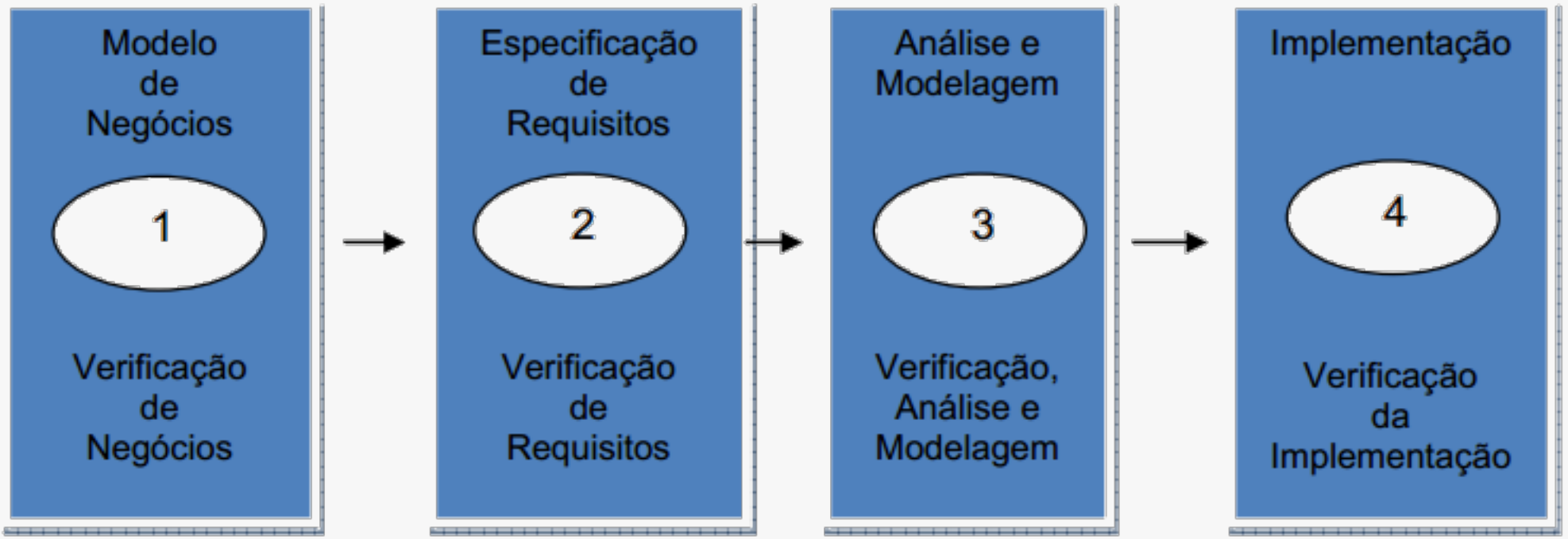
- A qualidade de todo o processo que está suportando o projeto de software,
- Valida se todas as documentações e atividades geradas estão dentro de um padrão desejável,
- Reduz os riscos de falhas de interpretação que seriam inseridas no produto de software no momento de sua construção.

Verificação de Software

Teste de Verificação

Os testes de verificação devem ser aplicados nas fases iniciais do processo de desenvolvimento.

Se os defeitos migrarem para as próximas fases, os custos de correção com certeza serão maiores.



Verificação de Software

Teste de Verificação

Realizamos verificações através das revisões e inspeções para garantir a qualidade das atividades e documentos produzidos em cada etapa do desenvolvimento de software.

Inspeções e revisões são termos comuns e bem familiares à maioria das pessoas. Isso porque não é incomum encontrarmos organizações que aplicam essas técnicas dentro de seu processo de desenvolvimento de software.

A revisão é um processo "humano" de análise de determinado documento. Esse processo requer pessoas adequadamente treinadas para desempenhar seus papéis durante a condução das atividades de verificação.

Essa técnica de teste de verificação pode ser realizada bem antes da execução do teste validação. Como cada fase gera seus próprios artefatos, estes devem ser analisados na medida em que são desenvolvidos a fim de evitar que artefatos defeituosos permaneçam no projeto.

Verificação de Software

Teste de Verificação

A inspeção é um processo formal baseado em regras e utilização de checklists. Durante a inspeção de software, os papeis dos participantes estão bem definidos.

Os revisores examinam a representação do sistema que pode ser um documento de requisitos, código fonte e outros.

Tem por objetivo encontrar defeitos nas fases iniciais do processo de desenvolvimento.

A utilização de checklists auxilia na uniformização e padronização do que será avaliado e dos padrões esperados em cada documentação. Esse instrumento orienta os trabalhos dos revisores e possibilita uma forma única de abordar os documentos e atividades a serem auditadas.

Através das revisões e inspeções, os defeitos são corrigidos antes que o artefato seja repassado para próxima fase do ciclo de desenvolvimento, com isso o custo de correção de erros é menor.

Verificação de Software

Teste de Verificação

Existem ainda as auditorias. São avaliações independentes feitas nos produtos de software, ou nos processos, a fim de averiguar com segurança a conformidade aos padrões, às diretrizes, às especificações e/ou aos procedimentos, incluindo documentos que especifiquem:

- A forma ou o conteúdo dos produtos;
- O processo pelo qual os produtos deverão ser produzidos.
- Como a conformidade aos padrões e às diretrizes deverá ser medida.

As auditorias têm por objetivo avaliar:

- Se em determinado projeto as diversas equipes estão respeitando o processo de desenvolvimento;
- Se estão registrando os defeitos;
- Se estão realizando as reuniões de revisões;
- Se estão realizando as documentações obrigatórias,
- Se estão envolvendo clientes e usuários nos processos e
- Se estão atualizando continuamente o mapa de riscos dos projetos.

Verificação de Software

Teste de Verificação

Geralmente realizamos verificações somente nos estágios iniciais do projeto, pois são as fases em que é gerado o maior número de documentos e registros de informações. No entanto, é natural que o projeto sofra a inserção de melhorias ou mesmo uma reestruturação de aspectos fundamentais do negócio, de modo a interferir nas estruturas e definições estabelecidas. As atividades de verificação devem ser executadas sempre que o projeto sofrer mudanças, de forma a manter a integridade das documentações.

Em processos de verificação pouco formais, é comum que após a execução de revisões nenhum documento seja gerado. Quando isso ocorre, perdemos parte da história das decisões do projeto de desenvolvimento.

O relatório de conclusão das verificações deve estabelecer os motivos da mudança de decisões e definições dos documentos, exatamente para podermos justificá-los e rastreá-los no futuro.

Validação do Software

Teste de Validação

Os testes de validação estão focados na garantia da qualidade do produto de software e têm por objetivo identificar o maior número possível de erros tanto nos componentes isolados quanto na solução tecnológica como um todo. O sucesso da validação está apoiado diretamente em um forte planejamento de todas as atividades de testes, nas quais a concentração dos trabalhos de validação nos componentes mais complexos e nos requisitos mais críticos contribui para aumentar a eficiência dos procedimentos de detecção de defeitos.

Os testes de validação envolvem a execução total ou parcial do software a ser avaliado, porém, existem duas únicas abordagens para conduzir um processo de validação: pela estratégia caixa branca ou pela estratégia caixa preta.

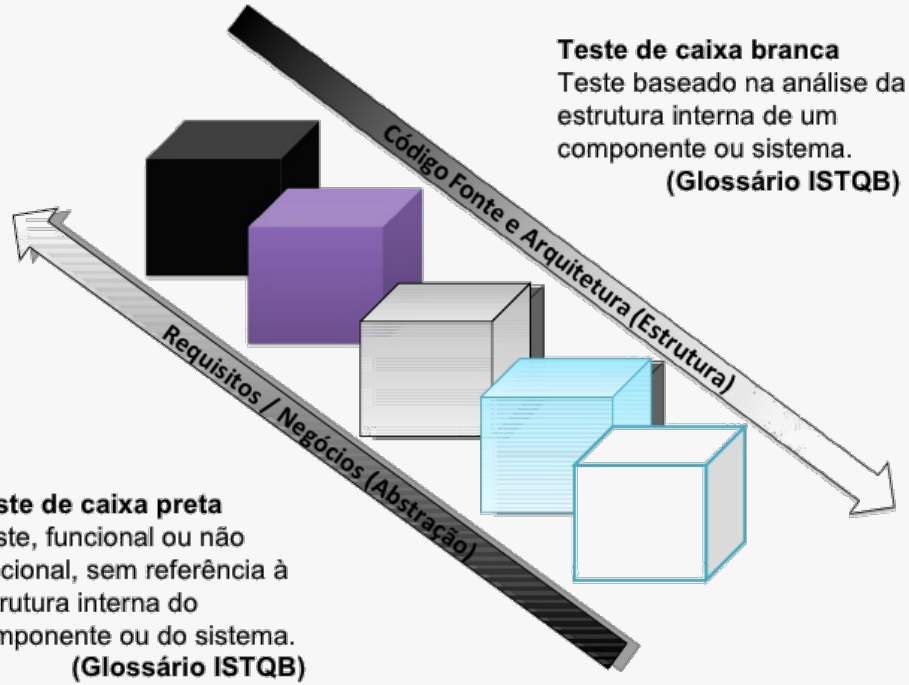
Nenhuma delas é completa, na realidade elas se completam e devem ser aplicadas em conjunto a fim de garantir um teste de boa qualidade.

Validação do Software

Teste de Validação

Os Testes de Caixa Preta são conhecidos por serem mais simples de se implantar do que os testes de caixa branca.

Na verdade, ambos são complexos e exigem grande esforço de planejamento e automação dos procedimentos, porém os testes de caixa preta são frequentemente encontrados nas organizações, em forma de testes manuais executados por profissionais ou mesmo usuários do sistema, o que facilita a introdução desse conceito nas organizações.



Teste de caixa branca
Teste baseado na análise da estrutura interna de um componente ou sistema.
(Glossário ISTQB)

Teste de caixa preta
Teste, funcional ou não funcional, sem referência à estrutura interna do componente ou do sistema.
(Glossário ISTQB)

Validação do Software

Teste de Validação – Caixa Preta

É o tipo de teste onde o programa é considerado uma entidade fechada e a sua estrutura interna não é considerada. Apenas as especificações do programa e os requisitos do software são considerados no momento da execução dos testes.

O teste caixa preta é também chamado de teste funcional. O termo "caixa preta" é usado também na engenharia e advém do fato de que, ao analisar o comportamento de um objeto, ignora-se totalmente sua construção interna.

O teste de caixa preta é baseado nos requisitos funcionais do software. Como não há conhecimento sobre a operação interna do programa, o avaliador se concentra nas funções que o software deve desempenhar. A partir da especificação são determinadas as saídas esperadas para certos conjuntos de entrada de dados. Esse tipo de teste reflete de certa forma, a ótica do usuário, que está interessado em se servir do programa sem considerar os detalhes de sua construção. Comparado a outros tipos de teste, este é relativamente mais simples.

Validação do Software

Teste de Validação – Caixa Branca

Os Testes de estrutura ou "caixa branca" são baseados na estrutura do software.

O teste de caixa branca, ao contrário do testes de caixa preta, é aquele em que o analista tem total acesso à estrutura interna da entidade sendo analisado. O teste caixa branca também é conhecido como teste estrutural e permite, por exemplo, que o analista escolha partes específicas de um componente para serem testadas.

O teste caixa branca permite ao avaliador concentrar a atenção nos pontos mais importantes ou "perigosos" do código, de uma maneira mais precisa do que no caso do teste caixa preta. Tais pontos podem até ser identificado durante o desenvolvimento e cobertos com uso de assertivas e testes.

O leitor agora deve pesquisar no google sobre testes de caixa branca e caixa preta para ter uma visão mais profundas das técnicas envolvidas nesses testes.

Dimensões do Teste

Introdução

A realização de testes é, quase sempre, limitada por restrições de cronograma e orçamento; eles determinam quantos testes será possível executar.

É importante que os testes sejam bem planejados e desenhados, para conseguir-se o melhor proveito possível dos recursos alocados para eles.

Podemos separar os testes em quatro dimensões:

1 – Estado do Teste
2 – Técnica do Teste
3 – Metas do Teste
4 – Ambiente do Teste

Vejamos mais sobre cada uma dessas dimensões.

Dimensões do Teste

Dimensão 1 – Estado do Teste

Momento onde haverá teste e o responsável por sua execução. Podemos considerar quatro tipos básicos:

- Teste de Unidade
- Teste de Integração
- Teste de Sistema
- Teste de Aceitação

Teste de Unidade

Estágio mais baixo da escala de testes e são aplicados nos menores componentes de código criados. Os testes unitários verificam o funcionamento de um pedaço do sistema ou software isoladamente ou que possam ser testados separadamente, podendo, inclusive, ser um programa ou um componente.

Têm por objetivo verificar um elemento que possa ser logicamente tratado como uma unidade de implementação.

Dimensões do Teste

Dimensão 1 – Estado do Teste

Teste de Unidade

Os testes unitários devem ser feitos pelos programadores e devem garantir que o programa está funcionando adequadamente.

Existem algumas ferramentas de automação que facilitam esse tipo de teste e que o tornam mais afetivo e rápido.

Normalmente essas ferramentas são específicas para a linguagem de programação usada.

Teste de Integração ou Iteração

É o Teste de Integração de componentes que já tenham sido passados pelo teste unitário, onde são executados numa combinação para verificar se eles funcionam corretamente juntos.

Tem por objetivo garantir que os componentes da aplicação, ou daquele módulo da aplicação, possam ser integrados com sucesso para executar uma determinada funcionalidade.

Dimensões do Teste

Dimensão 1 – Estado do Teste

Teste de Sistema

Teste de sistema se refere ao comportamento de todo do sistema/produto definido pelo escopo de um projeto ou programa de desenvolvimento. O teste de sistema deve garantir que os requisitos do software foram cumpridos e foram implementados corretamente. Nos testes de sistema é preciso a garantir que os usuários não irão encontrar erros grosseiros quando iniciarem o seu trabalho de homologação.

São realizados pela equipe de testes, visando a execução do sistema como um todo ou um subsistema (parte do sistema), dentro de um ambiente operacional controlado, para validar a exatidão e perfeição na execução de suas funções.

Esse estágio só será iniciado quando os testes de integração forem dados como encerrado. Será simulada a operação do sistema, sendo testadas todas as suas funções de forma mais próxima possível do que irá ocorrer no ambiente de produção.

Dimensões do Teste

Dimensão 1 – Estado do Teste

Teste de Sistema - Tipos

- Teste de Carga: Mede o comportamento de um componente ou sistema por meio do aumento de carga, por exemplo, número de usuários paralelos e/ou número de transações para determinar qual o tamanho de carga que pode ser suportada.
- Teste de Performance: Grau até o qual um sistema ou componente cumpre com as funções a ele destinadas de acordo com restrições de tempo de processo e de taxa de rendimento/throughput.
- Teste de usabilidade - Teste que determina a extensão até a qual o produto de software é entendido, fácil de aprender, fácil de operar e atraente para os usuários sob condições especifica.
- Teste de compatibilidade - Processo pelo qual testes são realizados para determinar o atendimento de um componente ou sistema.
- Teste de Segurança - Teste que determina a segurança de um produto de software.
- Teste de recuperação - Processo que determina a recuperabilidade de um produto de software.

Dimensões do Teste

Dimensão 1 – Estado do Teste

Teste de Aceitação

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

Objetiva estabelecer a confiança no sistema, parte do sistema ou uma característica não específica do sistema. Procurar defeitos não é o principal foco do teste de aceitação.

Teste de aceite freqüentemente é de responsabilidade do cliente ou do usuário do sistema; os interessados (stakeholders) também podem ser envolvidos.

Os testes, embora sejam de responsabilidade dos usuários, são conduzidos com suporte da equipe de testes e da equipe do projeto.

Dimensões do Teste

Dimensão 2 – Técnica do Teste

Fazer uma análise da aplicação e dizer qual técnica é mais apropriada para ser seguida. Vejamos algumas técnicas.

Técnica de Teste Estrutural

Teste baseado na análise da estrutura interna de um componente ou sistema. Teste estrutural deve ser baseado na arquitetura do sistema, como uma hierarquia de chamadas.

Podemos aplicar essa técnica em todos os níveis de teste

Técnica de Teste Funcional

Técnicas baseadas em especificação que podem ser utilizadas para derivar as condições de teste e casos de testes a partir da funcionalidade do software ou sistema.

Testes funcionais são baseados em funções, descritas nos seguintes documentos: especificação de requisitos; casos de uso, especificação funcional, ou podem não estar documentados. As funções representam "o que" o sistema faz.

Dimensões do Teste

Dimensão 3 – Metas do Teste

Fazer uma análise do projeto e informar quais tipos de testes será executado. É uma das mais fortes das três dimensões, mais conhecida como "tipo de teste".

Para os testes estruturais temos os seguintes tipos:

- Estresse: Determinar o desempenho do sistema com volumes esperados.
- Execução: Sistema funciona com proficiência (competência).
- Contingência: Se o sistema retorna a um status operacional depois de uma falha.
- Operação: Sistema pode ser executado em um status operacional normal.
- Conformidade: Se o sistema foi desenvolvido em concordância com padrões e procedimento.
- Segurança: Se o sistema esta protegido nos padrões de segurança da empresa.

Dimensões do Teste

Dimensão 3 – Metas do Teste

Para os testes funcionais temos os seguintes tipos:

- Requisitos: Sistema executado conforme especificado.
- Regressão: Verifica se alguma coisa mudou do que já estava funcionando corretamente.
- Erro no Manuseio (Usabilidade): Erros podem ser prevenidos ou detectados e depois concertados.
- Manual de Suporte: Preparação de dados para o uso de dados vindo do sistema.
- Integração: O Teste Intersistema é feito para assegurar que esta interconexão entre as aplicações funciona corretamente.
- Controle: Olhar negativo para o aplicativo assegurando que o que pode sair errado foi adequadamente protegido.
- Paralelo: Comparar resultados do aplicativo atual com a versão anterior.

Dimensões do Teste

Dimensão 4 – Ambiente do Teste

Nessa dimensão definimos o ambiente onde as outras três dimensões irão rodar, observando as evoluções tecnológicas e fazendo um bom planejamento da fase de teste. O ambiente é tudo que está "em volta do teste" no mundo real. Um ambiente de teste é composto por dois grandes grupos:

Elementos centrais de teste em si:
- Equipamento disponível para teste;
- Software e hardware disponíveis para teste;
- Recursos humanos disponíveis para interagir ou realizar o teste.

Elementos de apoio:

- Espaço físico para o trabalho de teste;
- Equipamentos de apoio ao teste;
- Ferramentas de apoio, como ferramentas de monitoração;
- Tecnologias de apoio;
- Material de apoio, treinamento, etc.

Processo de Teste

Introdução

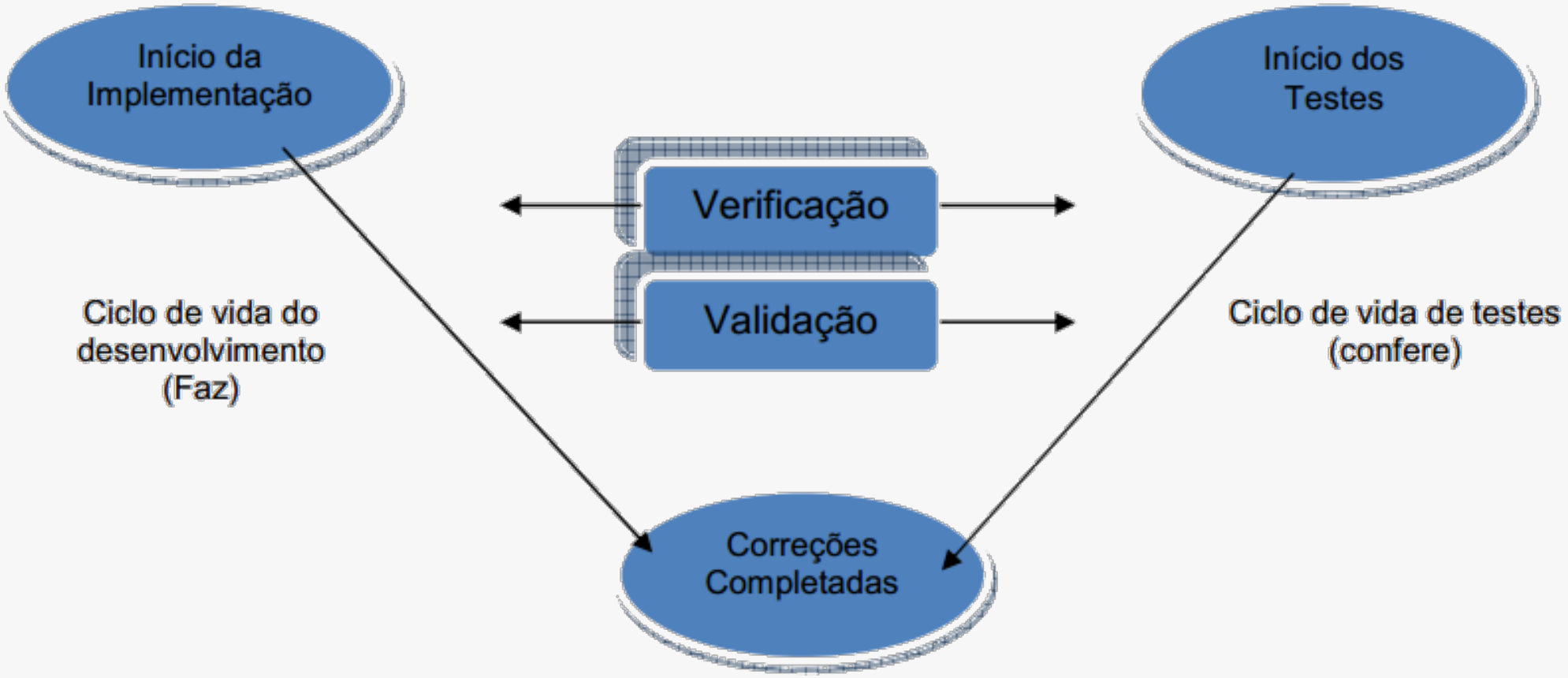
O processo de teste compreende fundamentalmente o planejamento, a especificação, a execução, o registro, a verificação da finalização e as atividades de fechamento. O Modelo "V" é uma estrutura que descreve as atividades do ciclo de vida do desenvolvimento de um software, desde a especificação de requisitos até a manutenção.

O modelo V ilustra como as atividades de teste podem ser integradas em cada fase do ciclo de vida do desenvolvimento de um software.

O processo de teste de software segue o conceito “V”, parte esquerda do “V” representada o ciclo de desenvolvimento de software e a parte direita, alguns dos passos do processo de teste de software. Os testes devem ser executados em todas as etapas do ciclo de vida do processo de desenvolvimento de software. Dentro deste enfoque, os testes são executados desde os requisitos até os testes de aceitação, na fase de homologar e liberar o software para a produção.

Processo de Teste

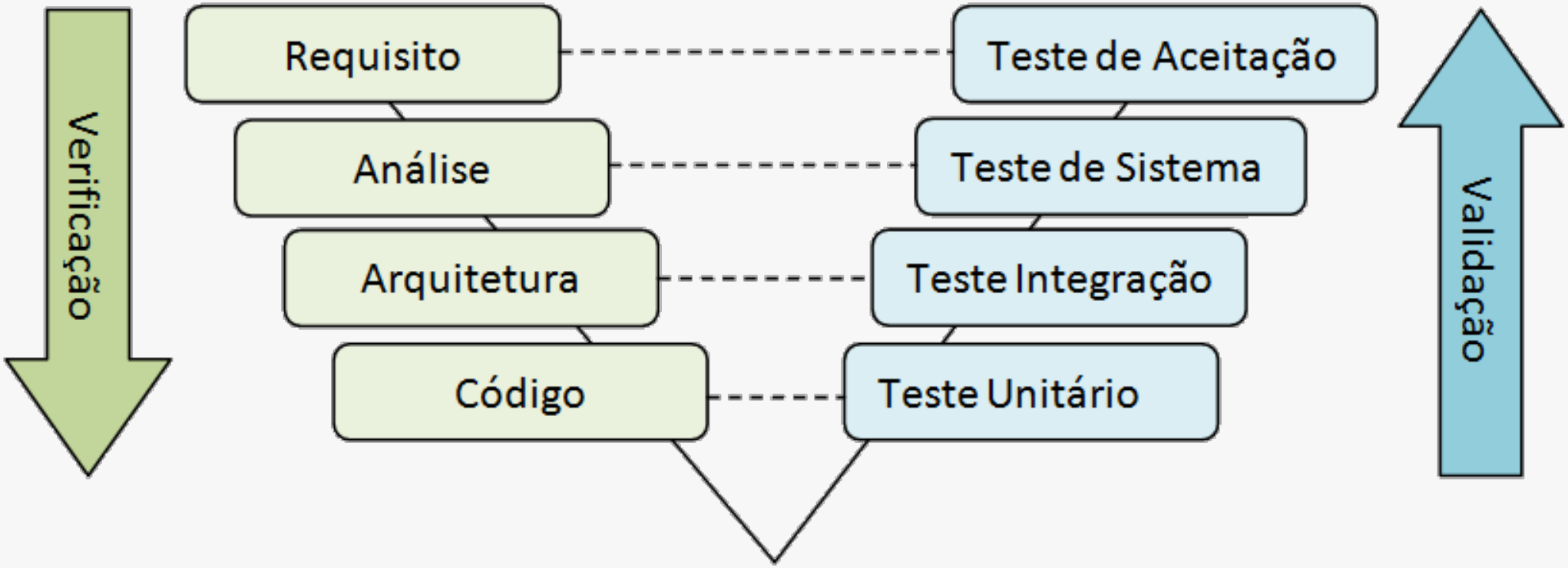
Modelo "V"



Conceito "V" de Teste de Software. Fonte: Bastos, Rios, Cristalli, Moreira 2006.

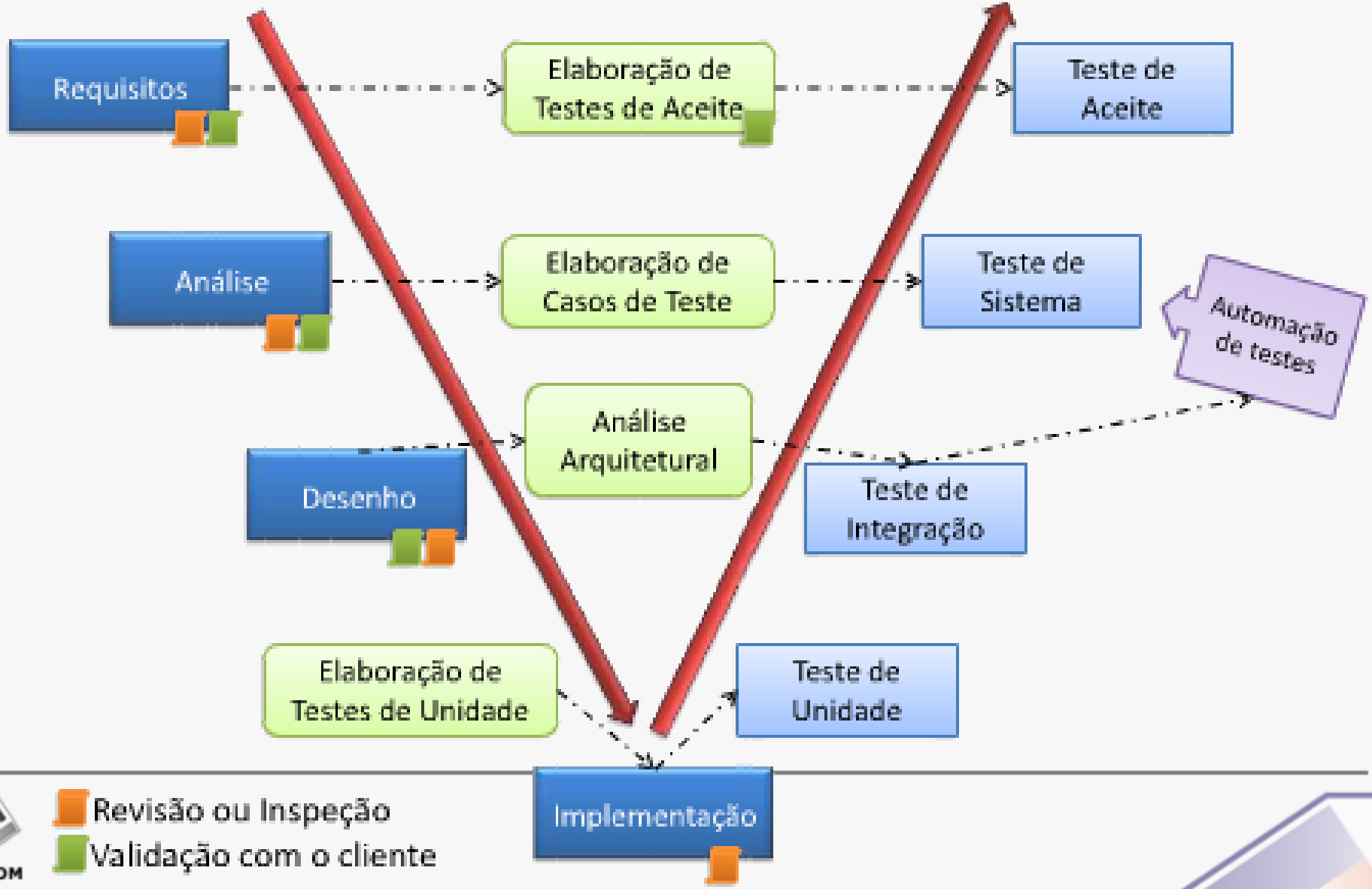
Processo de Teste

Modelo "V"



Processo de Teste

Modelo "V"



Processo de Teste

Profissional de Teste

Líder do Projeto de Testes	Técnico responsável pela liderança de um projeto de teste específico, normalmente, seja um projeto novo ou uma manutenção.
Arquiteto de Teste	É o técnico responsável pela montagem da infra-estrutura de teste, montando o ambiente de teste, escolhendo as ferramentas de teste e capacitando a equipe para executar o seu trabalho neste ambiente de teste.
Analista de Teste	É o técnico responsável pela modelagem e elaboração dos Casos de teste e pelos Scripts de Teste. Pode ser que em alguns casos os Scripts de Teste sejam elaborados pelos testadores.
Testador	Técnico responsável pela execução dos Casos de Teste e Scripts de Teste.

Processo de Teste

Considerações

O teste de software pode ser visto como uma parcela do processo de qualidade de software. A qualidade da aplicação pode e, normalmente, varia significativamente de sistema para sistema.

Quando a organização escolhe e mantém um processo de testes em que todos estão envolvidos, o software consequentemente terá menos defeitos e maior qualidade.

É importante que o leitor tenha a visão geral aqui apresentada e, se possível, se aprofunde no assunto.

Mesmo que o leitor trabalhe por conta própria ou numa pequena empresa, é interessante que elabore um processo mínimo de testes para evitar dores de cabeça durante a implantação e manutenção do sistema.

Referências

JESSE J. Garrett. Ajax: A new approach to web applications, 2005. Disponível em http://adaptivepath.org/ideas/ajax-new-approach-web-applications/. Acesso em Abril/2016.

BARTIE, Alexandre. Garantia da Qualidade de Software Campus.

BASTOS, Anderson; RIOS, Emerson; CRISTALLI, Ricardo; MOREIRA, Trayahu. Base de Conhecimento em Teste de Software. Traço & Photo.

KOSCIANSKI, André; SOARES, Michel dos Santos. Qualidade de Software: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software – 2ª EDIÇÃO - NOVATEC

MOLINARI, Leonardo. Teste de Software: Produzindo sistemas melhores e mais confiáveis, Érica.

MOLINARI, Leonardo. Testes Funcionais de Software, Visual Books.

PRESSMAN, R. Engenharia de Software - Makron Books.

RIOS, Emerson. Dissecando o padrão IEEE 829: Documentação de Teste de Software, Art Studio Editora.

RIOS, Emerson; TRAYAHU R. M. Filho Teste de Software - Alta Books.

SYLLABUS. Base de Conhecimento Para Certificação de Teste – Comissão Internacional para Qualificação de Teste de Software.

Wikipedia - https://pt.wikipedia.org. Acessos em Abril/2016.