Testes de Software para Web

Fundamentos do Teste de Software

MSc. Jonathas Santos

Legendas

Termo ou assunto que será abordado com detalhes em nas próximas aulas;

Termo ou assuntos que valem uma pesquisa posterior de vocês;

Um pouco sobre mim

Me contem sobre vocês!

Contextualização

Você é capaz de construir um software perfeito?

Fábula do avião...

Imagine que a Flexpeak presenteia você com uma viagem pro Caribe, com tudo pago! Passagem de 1ª classe e tudo que tem direito!

Fábula do avião...

Daí chega o grande dia!

Hora de pegar as malas, fazer o *checkin*, entrar na fila do embarque...

Você vai chegando perto da entrada do avião, e se depara com uma mensagem na porta:

Fábula do avião...

“Este avião possui um software de última geração para o controle de voo, que possui 10.000 linhas de código.

E o melhor, dessas 10.000, conseguimos executar 9.990!”

Fábula do avião...

A pergunta que não quer calar: você entraria nesse avião?

E se em uma dessas 10 linhas, for a chamada dessa função: `getCoordenadas()`**?**

Então, deve ser importante testar...

Fonte: https://br.pinterest.com/pin/331296116333464742/


NA REUNIÃO
DE HOMOLOGAÇÃO
TESTE
SOFTWARE
DEPRESSÃO
O USUÁRIO FAZ UM TESTE
QUE EU NUNCA FIZ

Conceitos de Teste

Teste de Software - Conceito

“Teste pode ser usado apenas para mostrar a presença de defeitos mas nunca sua ausência.” ***Dijkstra (1970)***

“Teste é o processo de executar um programa ou sistema com a intenção de encontrar erros.” ***Myers (1979)***

“Teste é uma atividade objetivando avaliar um atributo de um programa ou sistema. É uma medida de qualidade de software”. ***Hetzel (1988)***

Teste de Software - Conceito

“É apenas uma amostragem”. ***Roper (1994);***

“É um processo de engenharia concorrente ao processo de ciclo de vida do software, que faz uso e mantém artefatos de teste usados para medir e melhorar a qualidade do produto de software sendo testado”. ***Craig e Jaskiel (2002);***

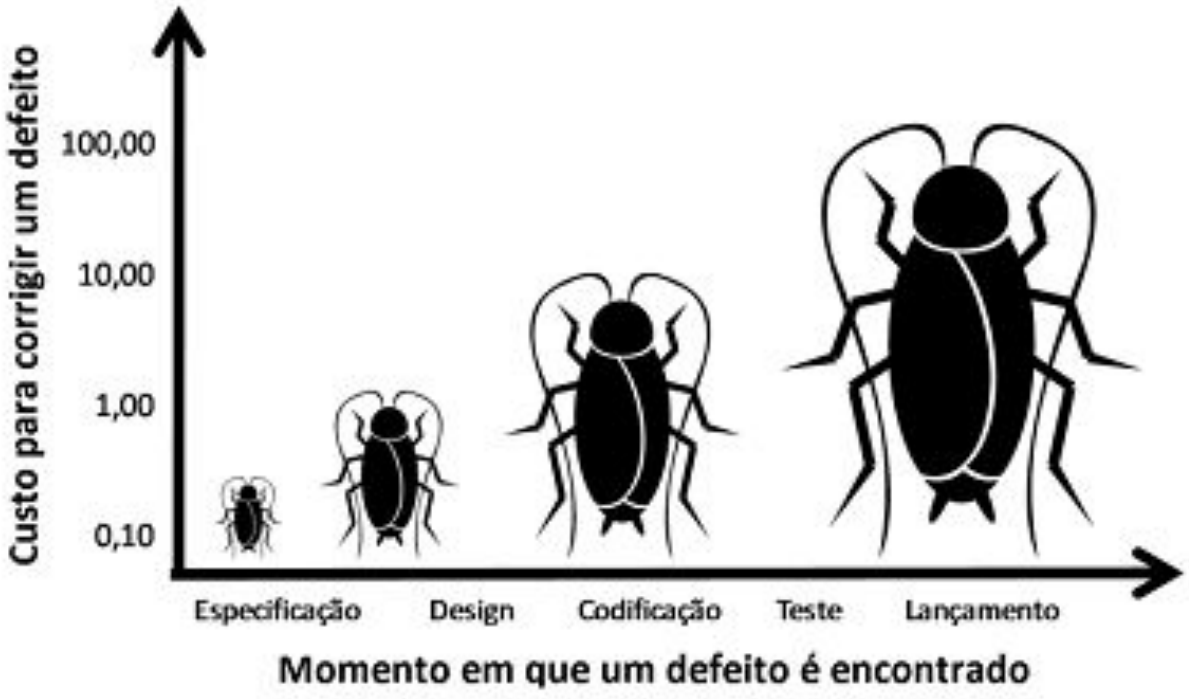
É o processo de executar um sistema ou componente sob condições específicas, observando e registrando os resultados, avaliando alguns aspectos do sistema ou componente.” ***(ISO/IEC/IEEE, 2010);***

Teste de Software - Por que fazer?

Crescente interesse e importância do teste de software, principalmente devido a demanda por produtos de software de alta qualidade;

Shull et al. (2002) alerta que quase não existem módulos livres de defeitos durante o desenvolvimento, e após a liberação dos mesmos, em torno de 40% podem estar livres de defeitos;

Boehm e Basili (2001) também apontam que é quase improvável liberar um produto de software livre de defeitos;



Além disso, quanto mais tarde um defeito é revelado, maior será o custo para sua correção!

São Francisco sem metrô por um dia

Falha no software de metrô no Brasil

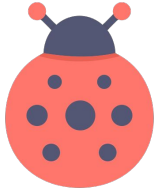
“Sequestro” virtual de um hotel inteiro

Terminologia e Conceitos Básicos de Teste

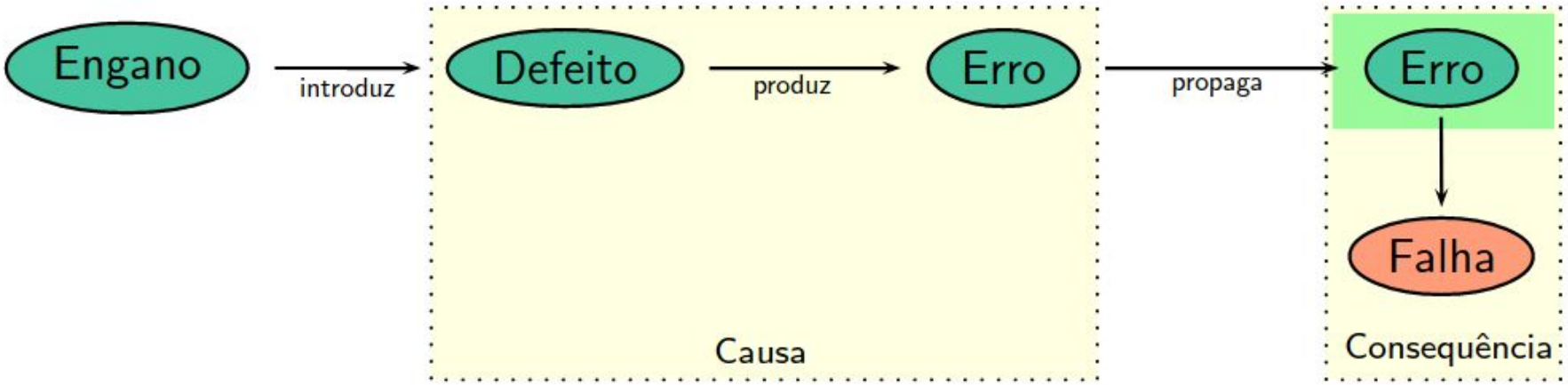
Terminologia de Defeitos

A ISO/IEC/IEEE 24765:2010 Systems and Software Engineering diferencia os seguintes termos:

- Engano (***Mistake***) – ação humana que produz um resultado incorreto;
- Defeito (***Fault***) – um passo, processo, ou definição de dados incorreta em um produto de software. No uso comum, os termos “erro”, “bug”, e“defeito” são usados para expressar esse significado;
- Erro (***Error***) – diferença entre o valor computado, observado ou medido e o valor teoricamente correto de acordo com a especificação;
- Falha (***Failure***) – inabilidade do sistema ou componente realizar a função requerida, considerando as questões de desempenho exigidas.



Termo usado na indústria para definir um **defeito/erro**. Sua origem é de 1976, quando um operador de navios encontrou uma mariposa dentro dos fios da máquina, o que pode ter sido a causa da falha no computador de bordo do navio.



Fluxo de Propagação do Erro. Fonte: Vincenzi, Delamaro & Maldonado

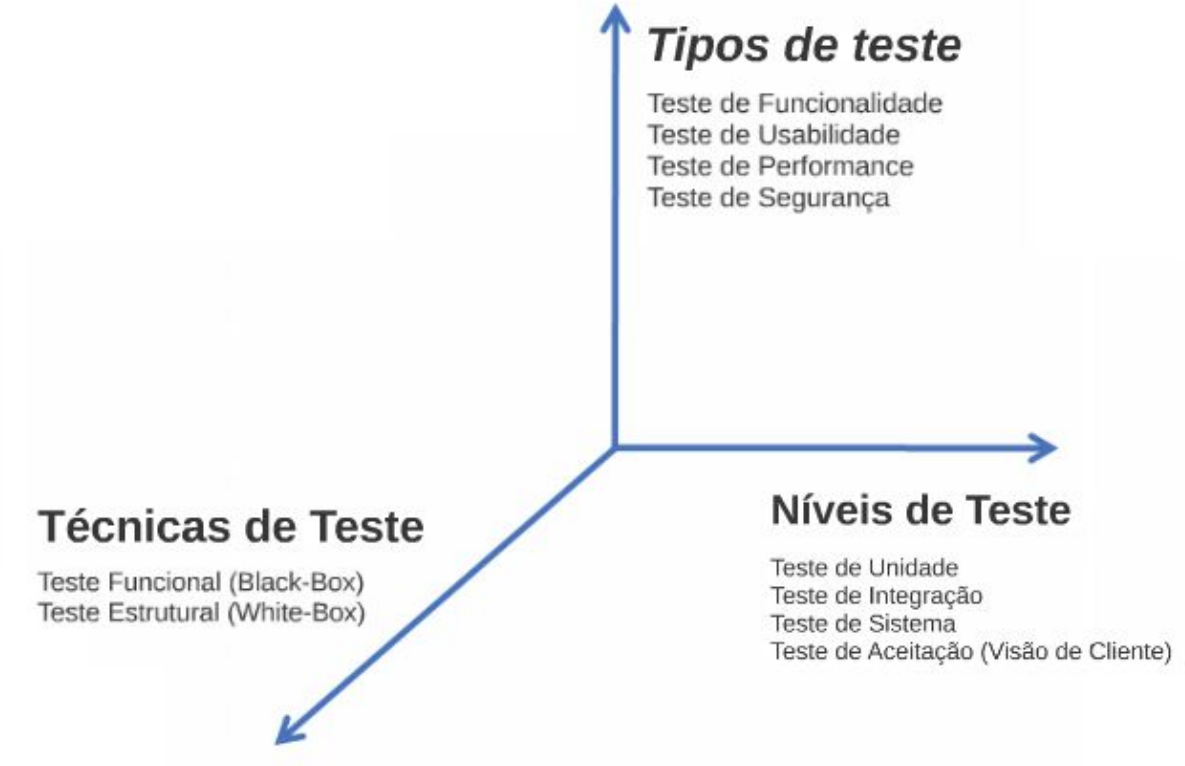
Terminologia

Considere o seguinte comando de atribuição: ***(z = y + x)***

1. **Engano** – o comando é modificado por *(z = y − x)*, caracterizando um defeito;
2. **Defeito** – se ativado (executado) com x = 0, nenhum resultado incorreto é produzido. Para valores de x diferentes de 0, o defeito é ativado causando um erro na variável z.
3. **Erro** – o estado errôneo do sistema, quando propagado até a saída, causará uma **Falha**

Tipos de Teste

Teste é um universo...



Teste é um universo...

Diferentes **tipos de testes** podem ser utilizados para verificar se um programa se comporta como o especificado;

Esses tipos são aplicados em diferentes **níveis de teste**, de acordo com a especificação do sistema;

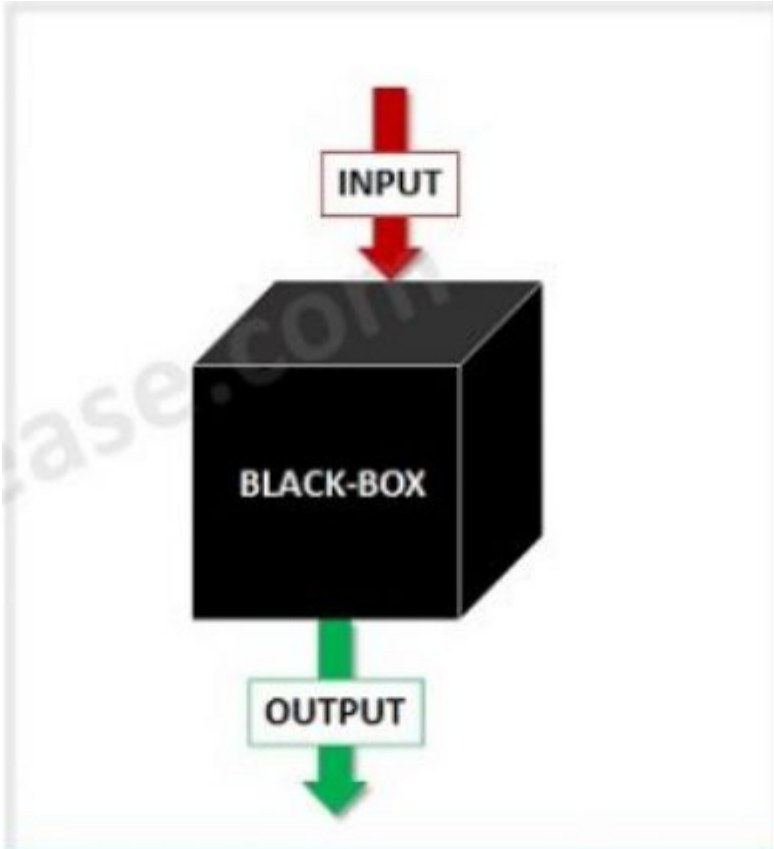
Esses tipos de teste utilizam **técnicas de teste** para gerar (ou escolher) os dados de entrada desses testes;

Técnicas de Teste

Técnica Funcional (caixa-preta)

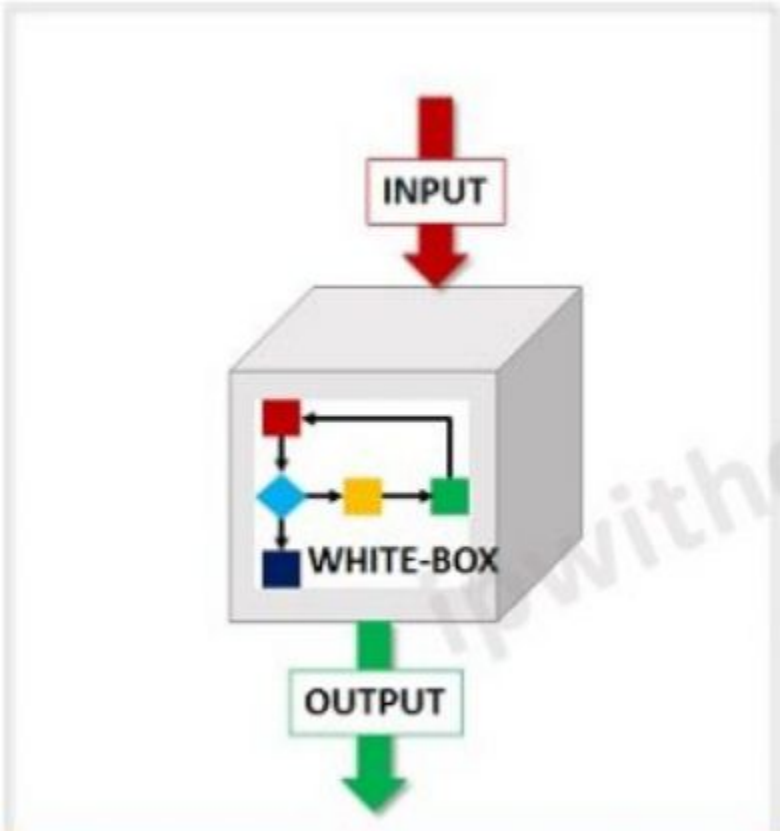
Os testes são baseados exclusivamente na especificação de requisitos do programa/componente em questão;

Nenhum conhecimento de como o programa/componte está implementado é requerido;



Técnica Estrutural (caixa-branca)

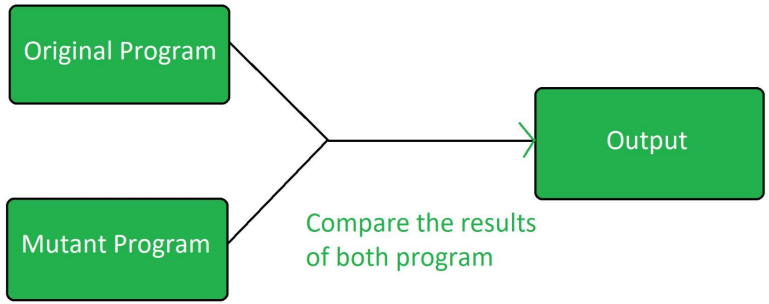
Os testes são baseados na estrutura interna do programa, ou seja, na implementação do mesmo.



Técnica baseada em Defeitos

Os testes são baseados em erros cometidos por desenvolvedores, normalmente;

Gera-se uma cópia do programa com um defeito proposital e executa os casos de teste para o programa;



Níveis do Teste

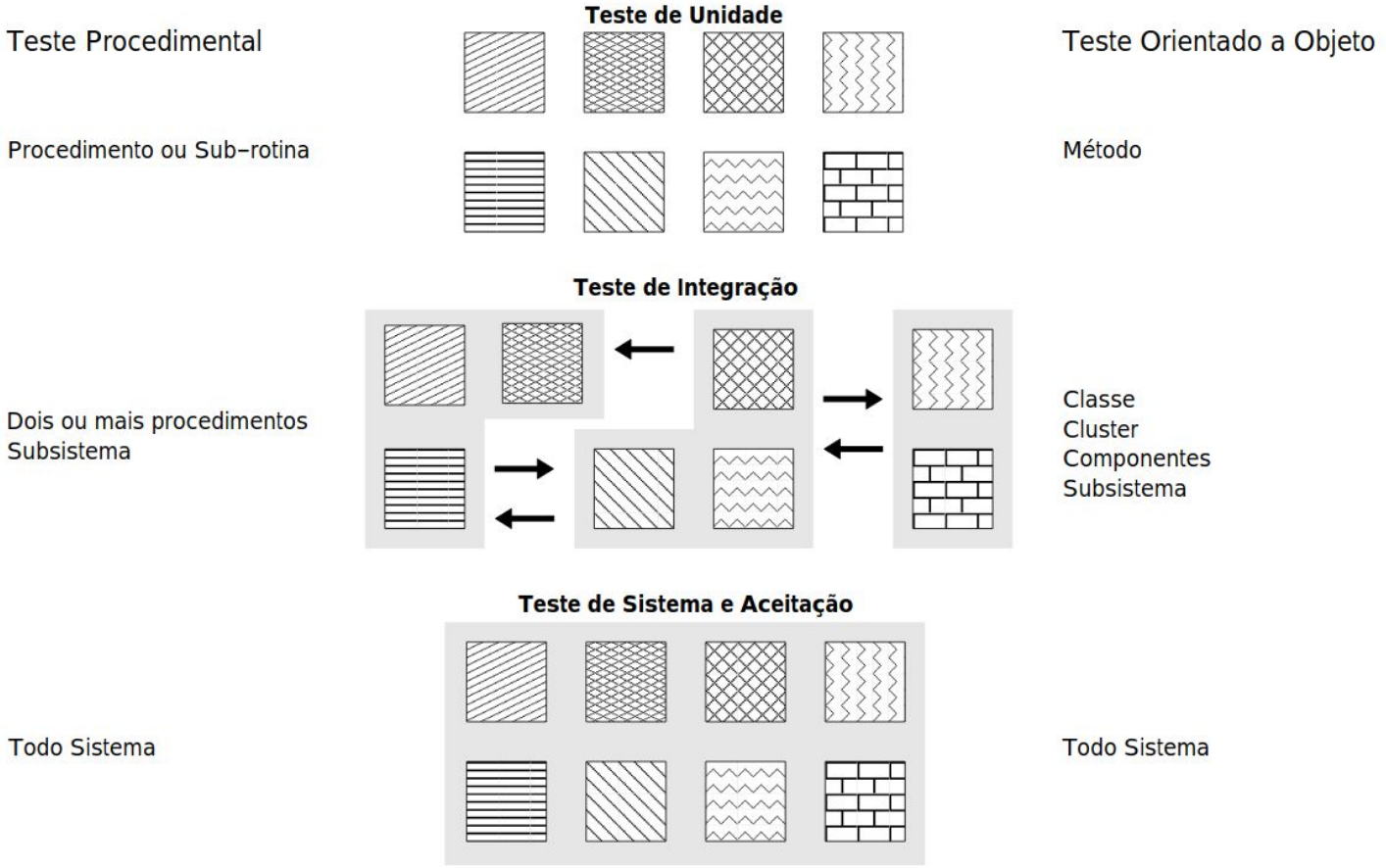
Níveis de Teste

A atividade de teste também é dividida em níveis ou fases, conforme outras atividades de Engenharia de Software;

O objetivo é reduzir a complexidade dos testes;

Conceito de ***"dividir e conquistar"***.

Começar o teste da menor unidade executável até atingir o programa como um todo.



Níveis do Teste. Fonte: Vincenzi, Delamaro & Maldonado

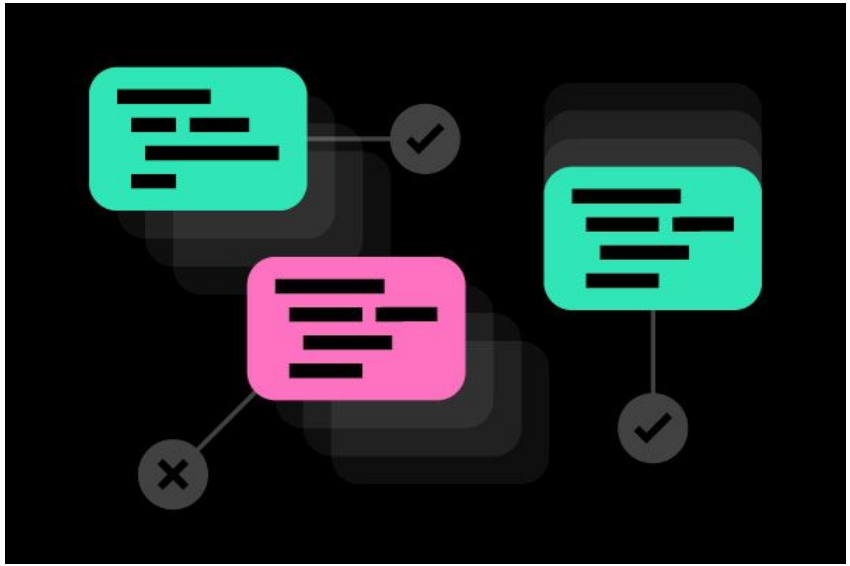
Teste de Unidade

Teste de Unidade

Objetivo é identificar erros de lógica e de programação na menor unidade de programação;

Diferentes linguagens possuem unidades diferentes:

- Pascal e C possuem procedimentos ou funções.
- Java, C++ e Python possuem métodos (ou classes);



Testes Unitários - Desafios

Como testar uma unidade que depende de outra para ser executada?

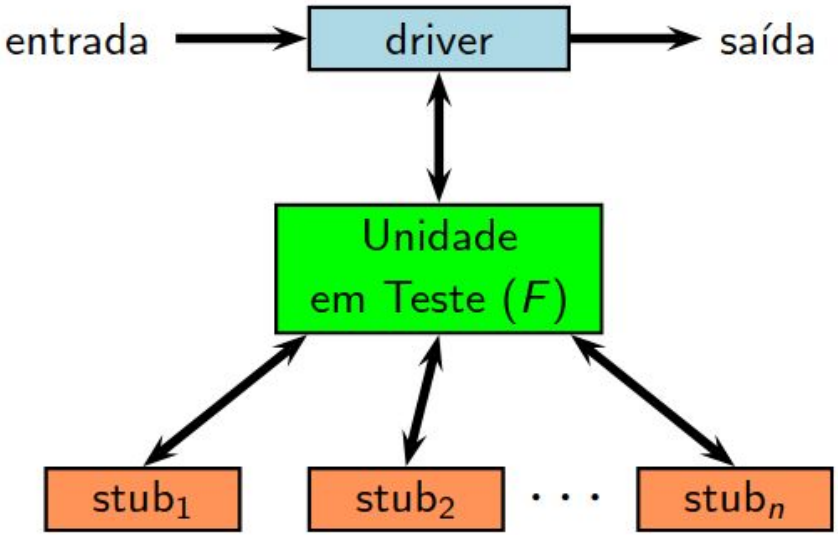
Como testar uma unidade que precisa receber dados de outra unidade para ser executada?

Para auxiliar no teste de unidade, em geral, são necessários ***drivers*** e ***stubs***.

Drivers e Stubs

O ***driver*** é responsável por fornecer para uma dada unidade os dados necessários para ela ser executada e, posteriormente, apresentar os resultados ao testador;

O ***stub*** serve para simular o comportamento de uma unidade que ainda não foi desenvolvida, mas da qual a unidade em teste depende;



Teste de Integração

Testes de Integração - Conceito

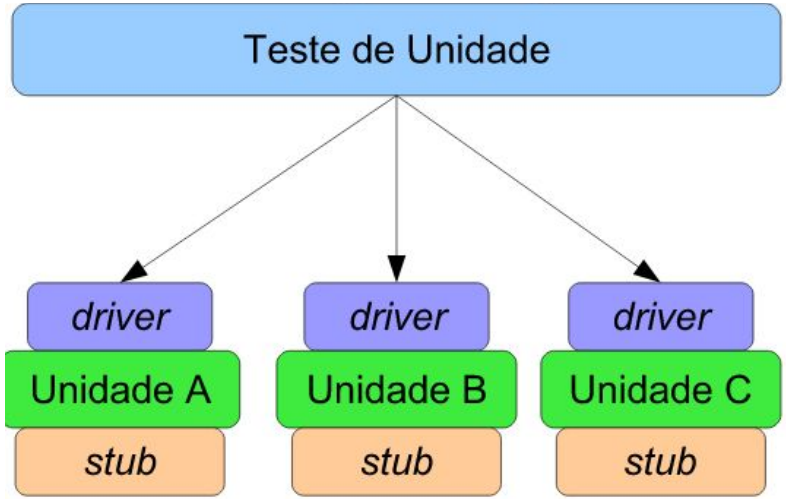
Objetivo é verificar se as unidades testadas individualmente se comunicam como desejado;

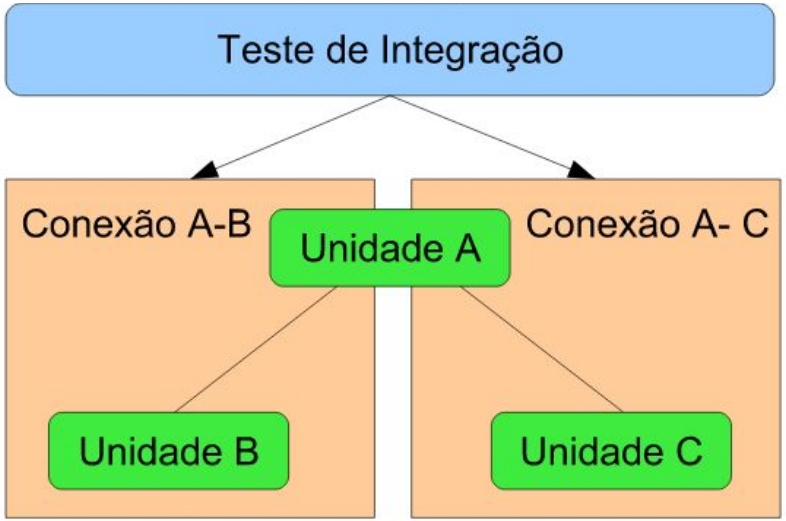
Por que testar a integração entre unidades se as mesmas, em isolado, funcionam corretamente?

Teste de Integração

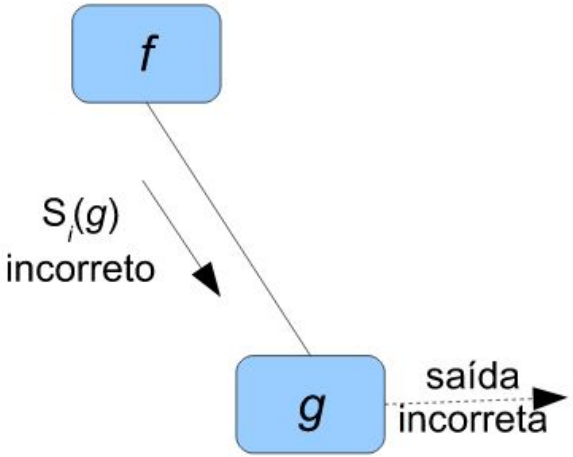
Dados podem se perder na interface das unidades;

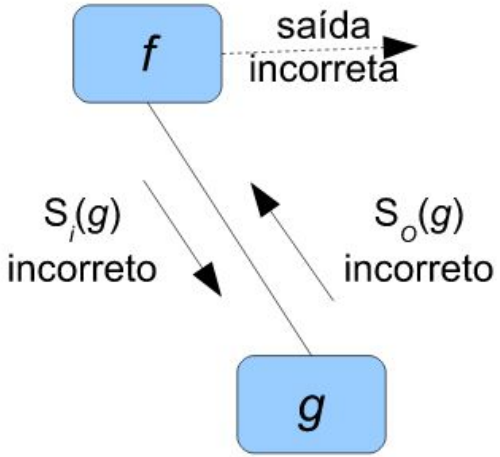
Variáveis globais podem sofrer alterações indesejadas;

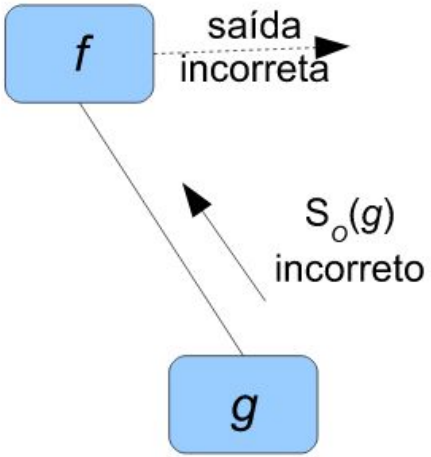




Teste de Integração - Tipos de Erros







Teste de Sistema

Teste de Sistema - Conceito

Verificar se o programa em si interage corretamente com o sistema para o qual foi projetado; Isso inclui, por exemplo, o SO, banco de dados, hardware, manual do usuário, treinamento, etc;

Corresponde a um teste de integração de mais alto nível;

Inclui teste de funcionalidade, usabilidade, segurança, confiabilidade, disponibilidade, performance, backup/restauração, portabilidade, entre outros (Norma ISO-IEC-9126 para mais informações (ISO/IEC, 1991))

Teste de Aceitação

Teste de Sistema - Conceito

Verificar se o programa em si interage corretamente com o sistema para o qual foi projetado; Isso inclui, por exemplo, o SO, banco de dados, hardware, manual do usuário, treinamento, etc;

Corresponde a um teste de integração de mais alto nível;

Inclui teste de funcionalidade, usabilidade, segurança, confiabilidade, disponibilidade, performance,backup/restauração, portabilidade, entre outros (Norma ISO-IEC-9126 para mais informações (ISO/IEC, 1991))

Teste de Sistema - Tipos

Teste Alfa

- Realizado pelo usú ario no ambiente do desenvolvedor;
- O desenvolvedor tem controle sobre o ambiente de teste e monitora as ações do usuário registrando falhas e problemas de uso;

Teste Beta

- Realizado pelo usuário no seu ambiente;
- O desenvolvedor não tem controle sobre o ambiente utilizado nos testes;

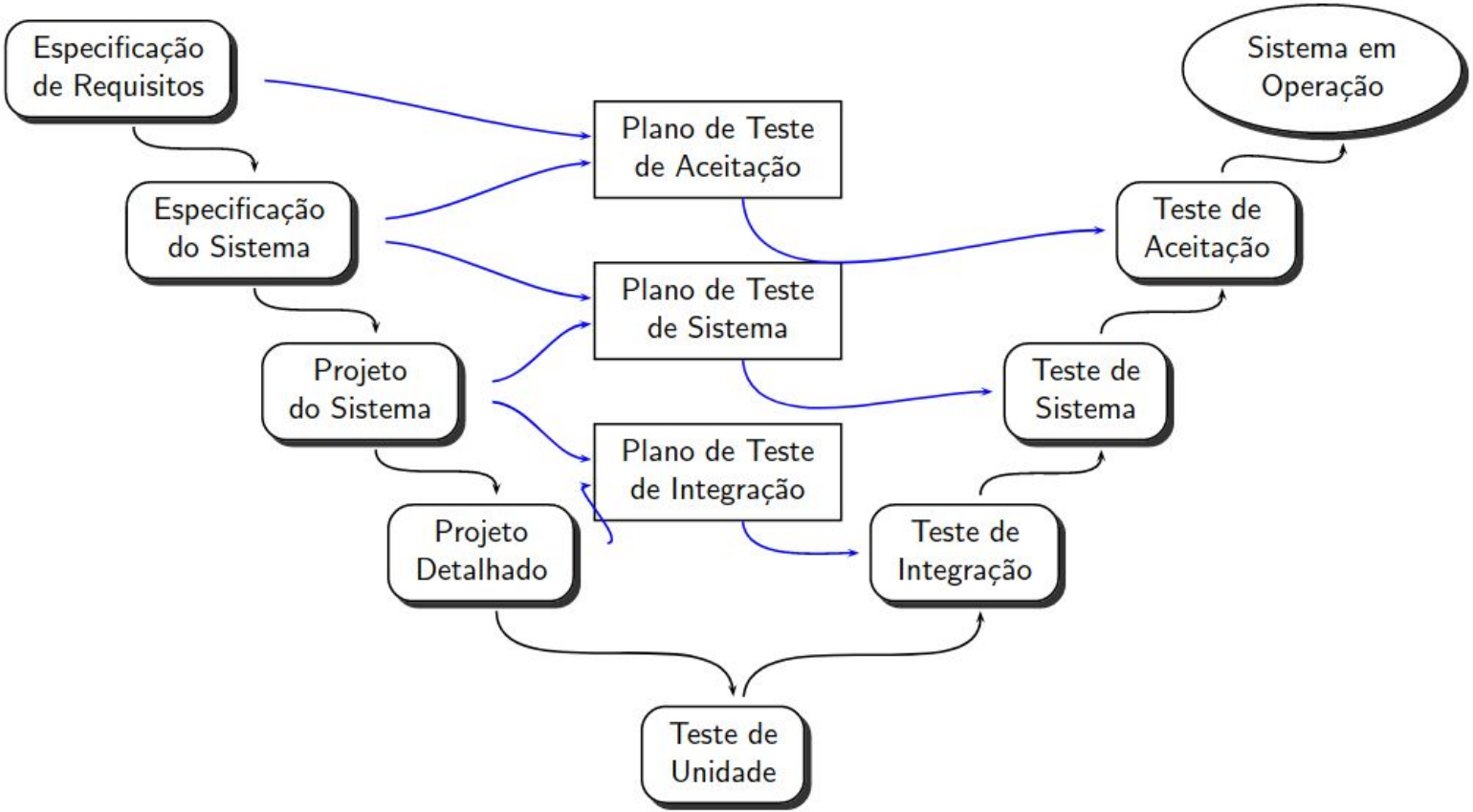
Teste e o desenvolvimento de software

Teste e o desenvolvimento de software

Em um dos conceitos de teste vistos anteriormente, teste de software é uma atividade do desenvolvimento;

Se observarmos atentamente, para chegarmos ao produto final, o software passa por várias etapas e suas funcionalidades vão se transformando e compondo diversos tipos de artefatos;

Pela crescente qualidade exigida dos produtos de software, há um conjunto de atividades que podem ser realizadas em cada fase desse ciclo;



Relação entre as fases de desenvolvimento e atividades de teste referentes a cada uma.

Teste de Software - Desafios

Alguém aí já testou um produto de software?

Alguns problemas comuns são:

1. Não há tempo para o teste exaustivo;
2. Muitas combinações de entrada para serem exercitadas;
3. Dificuldade em determinar os resultados esperados para cada caso de teste;
4. Requisitos do software inexistentes ou que mudam rapidamente;
5. Não há tempo suficiente para o teste;
6. Não há treinamento no processo de teste;
7. Não há ferramenta de apoio;
8. Gerentes que desconhecem o teste ou que não se preocupam com qualidade.

Teste de Software - Boas práticas

Boas práticas de teste

- Dispensar tempo para o teste e planejá-lo da melhor forma;
- Possuir testadores maduros em relação ao negócio;
- Definir a relevância das funcionalidades a serem testadas;
- Ser pessimista, com cuidado para não ignorar cenários aparentemente inocentes;
- Registrar todas as falhas encontradas e acompanhar até que seja corrigida;
- Executar de forma natural e não roboticamente, explorando quando necessário para fazer alguma mudança ou encontrar alguma falha no sistema que passou despercebida;

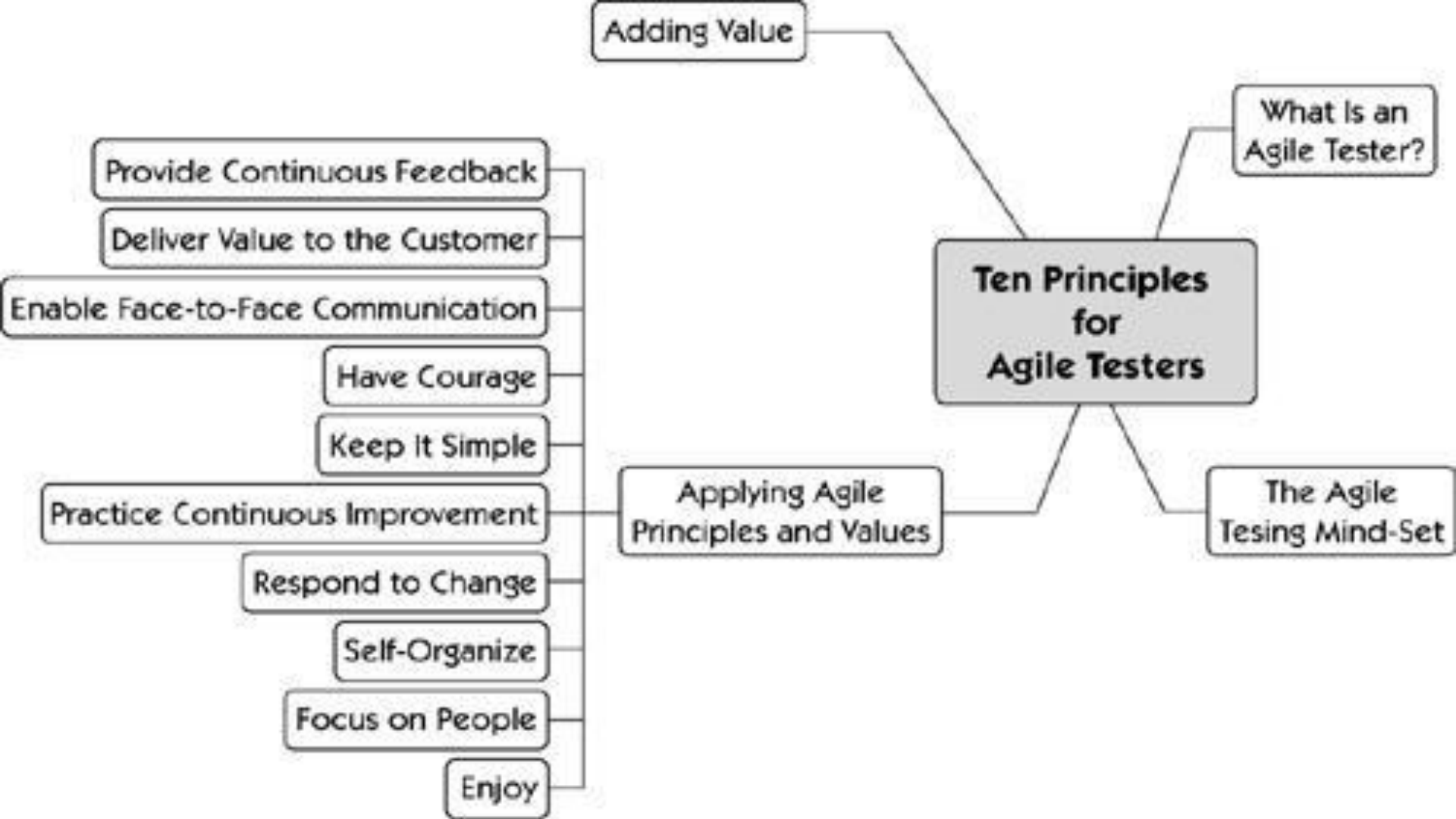
Boas práticas de teste

- Descrever ao máximo alguma falha encontrada facilitando o trabalho da equipe de desenvolvimento e obtendo todos os detalhes do que está errado e em que parte do sistema o bug ocorreu;
- Retestar para garantir que está tudo ok e verificar se o bug foi corrigido pelo desenvolvedor;
- Ao reportar uma falha, observar se o ocorrido com o produto está relacionado apenas a tal produto ou se envolve produtos da mesma categoria, cor, faixa de preço, etc;

Princípios do Teste Ágil

Princípios dos testes ágeis

- **Todos** em um time ágil são testadores;
- **Qualquer um pode realizar tarefas de testes**;
- Se isso é verdade, então o que é especial em um testador ágil? Se eu me defino como um testador em um time ágil, o que isso realmente significa?
- Testadores ágeis precisam de um conjunto de habilidades diferentes daquelas empregadas em equipes tradicionais? O que os guiam em suas atividades diárias?


Adding Value
What Is an Agile Tester?
Ten Principles for Agile Testers
Provide Continuous Feedback
Deliver Value to the Customer
Enable Face-to-Face Communication
Have Courage
Keep It Simple
Practice Continuous Improvement
Respond to Change
Self-Organize
Focus on People
Enjoy
Applying Agile Principles and Values
The Agile Tesing Mind-Set

O que é um testador ágil?

“Um profissional que aceita as mudanças, que colabora com o pessoal do negócio quanto da TI, e entende a concepção de usar os testes visando documentar requisitos e direcionar o desenvolvimento.”

Testador ágil

- Tendem a ter boas habilidades técnicas
- Sabe do seu papel ao colaborar na automação de testes
- São experientes em testes exploratórios
- Estão dispostos a aprender o que o cliente faz para então melhor entender quais são os requisitos de negócio

A mentalidade dos testadores ágeis

- É o que foca continuamente em fazer o melhor trabalho e entregar o melhor produto possível;
- Envolve disciplina, aprendizado, tempo, experimentações e trabalho conjunto com o time de desenvolvimento;
- Projetos bem sucedidos são o resultado de boas pessoas que se permitem fazer um bom trabalho;

A mentalidade dos testadores ágeis

- Não se vê como um policial da qualidade, que está tentando defender seus clientes de códigos inadequados
- Ele está pronto para reunir e compartilhar informações, de forma que os produtos entregues expressem realmente aquilo que foi pedido pelo cliente, e prover feedback do progresso do projeto

A mentalidade dos testadores ágeis

- Um testador ágil não se intimida em participar de uma discussão no projeto na qual pode melhorar a testabilidade, ou de sugerir uma solução mais elegante
- A mentalidade ágil é aquela orientada por resultados, criatividade, colaborativa, vontade em aprender, obstinado a entregar valor em tempo hábil

Aplicando princípio e valores ágeis

- Valores e princípios ágeis promovem o foco nas pessoas envolvidas no projeto, e como elas interagem e se comunicam
- O manifesto ágil inclui uma lista de princípios que definem como abordar o desenvolvimento de software, e os princípios de testes ágeis são parcialmente variantes daqueles princípios.

helpper

Os 12 princípios ágeis

1. Priorize o Cliente

2. Mudanças são bem-vindas

3. Entregas que geram valor

4. Cooperação entre squads e stakeholders

5. Confie e apoie

6. Conversa face a face

7. Priorize o que funciona

8. Desenvolvimento incremental

9. Exelência técnica

10. Mantenha a simplicidade

11. Times autônomos e auto-gerenciavéis

12. Retrospectiva e planejamento

Prover feedback contínuo

- Provê a comunicação e é uma parte importante para as equipes ágeis
- O feedback ajuda o Product Owner ou o cliente a articular, junto aos testadores, os requisitos das histórias de usuário
- No caso de impedimentos e obstáculos, estes serão conhecidos e poderão ser removidos

Entregar valor ao cliente

- O cliente prioriza a entrega da interação
- O time avalia o caminho crítico
- O Cliente muitas vezes tem o desejo de adicionar recursos interessantes, mas o testador sabe reconhecer qual o impacto dessa adição na história
- É melhor se precaver para entregar o principal, do que ficar focando em recursos adicionais que podem ser incrementados futuramente

Comunicação contínua

- Nenhum time funciona bem sem uma boa comunicação
- O testador ágil deve buscar maneiras únicas para facilitar a comunicação
- Testadores ágeis enxergam as histórias sob o ponto de vista do cliente, mas também entendem aspectos técnicos relacionados à implementação
- Testadores podem ajudar o cliente e os desenvolvedores a conversarem em linguagem comum

Caso real

"Quando eu estava trabalhando com uma equipe, nós tivemos um problema real com os programadores conversando com o PO e deixando os testadores de fora da discussão. Eles quase sempre descobriam as mudanças após o fato. Parte do problema era que os desenvolvedores não estavam conversando com os testadores devido a problema logísticos. Outro problema é a história. A equipe de teste era nova, e o proprietário foi conversar direto com os programadores."

Caso real

"Eu levei o problema para a equipe, e criamos uma regra. Encontramos grande sucesso com o “Power of Three”. Isto significa que todas as discussões sobre uma entrega é necessário envolver o programador, o testador e o PO. Era responsabilidade de cada um verificar a presença de um representante do grupo. Se alguém viu duas pessoas conversando, ele tinha o direito de intrometer na conversa. Isto funcionou, não poderia considerar deixar o testador de fora da discussão. Isso funcionou para nós, porque a equipe comprou a solução"

Tenha coragem

- É um valor do XP
- Práticas como automação de testes e integração contínua permite praticar este valor
- Quando você tem a primeira experiência como time ágil ou quando se faz uma transição, é normal o “receio” e também ter uma lista de perguntas que precisam de respondidas

Tenha coragem

- Como seremos capazes de completar tarefas de testes para cada história em tão pouco tempo?
- Como manter-se à vontade com o desenvolvimento?
- Como saberemos que quantidade de testes é suficiente?
- Se de repente você se deparar com algum procedimento que para você não faz parte do ágil e ninguém tem a resposta...

Tenha coragem

- Precisamos se permitir falhar – podemos aprender com ela
- Aconteceu um problema? - Iremos começar a pensar em maneiras desse acontecimento não se repetir
- É preciso coragem para permitir que outras pessoas cometam erros
- Peça ajuda, faça perguntas, sugira!

Seja simples

- Kent Beck do XP aconselhou a fazer a coisa mais simples e que se possa trabalhar
- Simples não significa fácil – teste o suficiente, adote técnicas e habilidades que te permitam trabalhar
- Precisamos automatizar os testes de regressão
- **Testes de fumaça** – possível automatizar os testes voltados para o negócio
- **Testes exploratórios** – aprender sobre a aplicação e desentocar erros inencontráveis

Melhoria contínua

- Procurar formas de realizar o seu trabalho – o time inteiro deve pensar desta forma
- Testadores participam de retrospectivas, avaliam o que está funcionando bem, ou o que precisa ser adicionado ou ajustado
- Testadores e times ágeis estão sempre buscando novas habilidades, ferramentas ou práticas que o ajudarão a entregar valor ou ter um bom retorno sobre o investimento do cliente
- Testadores vão em reuniões e conferências, participam de discussões, ler blogs e artigos, sempre buscando novas ideias
- Procuram automatizar, de forma a ganhar mais tempo para contribuir com a sua valiosa experiência

Caso real

"Nossa equipe utilizou as retrospectivas para um grande benefício, mas sentimos que era preciso algo novo para nos ajudar a fazer um trabalho melhor. Sugeri que fosse mantido um “backlog de impedimento”, de forma a registrar o que nos impedia de sermos produtivos como gostaríamos de ser.

Caso real

"O primeiro registro foi de que o nosso ambiente de testes estava muito lento. Nosso administrador do sistema barganhou duas máquinas e as transformou em dois novos servidores.

Nosso DBA analisou o desempenho do banco de dados de testes, e descobriu que o sistema de um disco era o impedimento e foi dado o sinal verde para instalar um RAID para que o acesso ao disco fosse melhorado. Logo fomos capazes de fazer deploys e conduzir nossos testes exploratórios de forma rápida".

Responda rápido a mudanças

- É a chave do valor para práticas ágeis
- É uma das mais difíceis concepções para testadores
- Mudanças constantes nos requisitos são um pesadelo, contudo, como testadores ágeis, nós devemos aceitar mudanças
- É uma decisão astuta de negócio que pode resultar em tempo desperdiçado se as histórias são repriorizadas ou se houverem mudanças em larga escala

Responda rápido a mudanças

- Testadores ágeis vão no fluxo e trabalham com o time de forma a acomodar as mudanças
- Testes automatizados é uma das chaves para a solução. Uma coisa temos certeza: nenhum time ágil será bem sucedido somente com testes manuais
- Nós precisamos de testes automatizados robustos de modo a entregar valor para o negócio a tempo construindo produtos valiosos

Auto organização

- Testadores ágeis é uma parte de times auto-organizados
- A cultura do time imbue o testador ágil a filosofia
- Quando o time (desenvolvedores, especialista de dados, cliente, analistas etc) pensa continuamente em testes de automatização, testadores apreciam um nova perspectiva

Auto organização

- Automatizar testes é difícil, mas é muito mais fácil quando você tem um time inteiro trabalhando juntos.
- Qualquer issue de testes é fácil de lidar – pessoas com múltiplos conhecimento atacando-as
- Quanto o time ágil encara um grande problema - é um problema de todos.
- As issues de prioridade alta – todo o time trabalha para solucionar

Foco em pessoas

- Projetos dão resultados quando boas pessoas estão dispostas a fazer o melhor trabalho
- Os valores e princípios ágeis foram criados com o objetivo de permitir o sucesso individual e de equipe
- Testadores que não se preocupam em aprender novas habilidades e crescer profissionalmente, contribuem à percepção de que o teste é um trabalho pouco qualificado
- Um testador exploratório hábil pode descobrir problemas no sistema que não pode ser detectada por testes funcionais automatizados

Divirta-se

- Trabalhar em um time onde todos colaboram, onde todo o time assume responsabilidades pela qualidade e testes, na nossa opinião, é algo que se pode alcançar
- Desenvolvimento ágil recompensa os testadores ágeis a terem paixão pelo seu trabalho
- Nosso trabalho como testadores é particularmente gratificante porque “nosso ponto de vista e habilidades nos permite adicionar real valor ao nosso time”

Soma de valores com o time

- O que esses princípios trazem para o time?
- Juntos eles trazem valor para o negócio
- O time usando diversos “chapéus”, e o desenvolvimento ágil tende a evitar que as pessoas sejam classificadas por especialidade
- Testadores ágeis não pensam apenas sob o ponto de vista dos clientes, mas também compreendem restrições técnicas e de implementação
- Se estão codificando os requisitos certos, os clientes estarão felizes

Adicionando valor

- Nas estimativas e sessões de planejamento, os testadores ágeis olham para cada recurso em múltiplas perspectivas: negócio, usuário final, suporte, programadores
- Ajudam a garantir que o cliente forneça requisitos claros e exemplos
- Até o fim da iteração, os testadores verificam se o mínimo de testes foi completado

Adicionando valor

- O time não precisa se identificar principalmente como testadores, mas verificou-se que equipes que se beneficiam das competências que os testadores profissionais têm desenvolvido, os princípios discutidos vão ajudá-los a fazer um bom trabalho de testes e a entregar valor

Dúvidas?

Referências

[1] BSTQB – Brazilian Software Testing Qualifications Board.
http://www.bstqb.org.br/uploads/docs/syllabus_2007br.pdf.

[2] INFO Online. “Defeito de Software põe doentes em perigo”.
http://info.abril.com.br/aberto/infonews/012009/16012009-19.shl. Publicado em 16/01/2009.

[3] G1. “Show de Madonna: fãs que tiveram cartão debitado terão ingresso”.
http://g1.globo.com/Noticias/Musica/0,,MUL749176-7085,00.html. Publicado em 05/09/2008.

[4] Base de Conhecimento em Teste de Software. 2ª Edição. Rios, Emerson; Cristalli, Ricardo; Moreira, Trayahú & Bastos, Aderson. – S. Paulo, Martins Fontes, 2007.

[5] Alexandre Bartie. “Processo de Teste de Software – Parte 01”.
http://imasters.uol.com.br/artigo/6102/des_de_software/processo_de_teste_de_software_parte_01/. Publicado em 07/05/2007.

Referências

[6] Gustavo Quezada. “Papéis e Responsabilidades na Equipe de Testes”. http://www.alats.org.br/Default.aspx?tabid=206&mid=1073&ctl=Details&ItemID=109. Publicado em 05/09/2008.

[7] Alexandre Bartie. Fábrica de Testes – Parte 01. http://imasters.uol.com.br/artigo/4435/des_de_software/fabrica_de_testes_parte_01/. Publicado em 26/07/2006.

[8] Roberto Murillo. A evolução do teste de software. http://imasters.uol.com.br/artigo/9369/des_de_software/a_evolucao_do_teste_de_software/. Publicado em 11/07/2008.

Read more: http://www.linhadecodigo.com.br/artigo/2775/introducao-ao-teste-de-software.aspx#ixzz6LsVlFi2k6