Introdução a testes e qualidade de software

João Paulo Menezes

- Analista de automação de testes na Zup/Itaú
- Gestor de Projeto e Instrutor na INexus Q.C
- Instrutor de testes e qualidade Freelancer

Linkedin
WhatsApp

Projetos que já atuei:

Conteúdo

Introdução a testes e qualidade de software

1. O que é QA.....
2. Qualidade ao longo do ciclo de desenvolvimento.....
3. O que é teste.....
4. O que é critérios de aceite .....
5. Fluxo do teste .....
6. Plano de testes e cenários/casos de testes.....
7. BDD (Behavior driver development).....

Hard Skills x Soft Skills

Hard Skills
Habilidades ensináveis ou conjuntos de habilidades fáceis de aprender

vs.

Soft Skills
Também conhecido como "habilidades pessoais" ou "habilidades interpessoais"

Falar uma língua estrangeira

Ensino superior

Rápida digitação

Operação maquinaria

Programação

Comunicação

Flexibilidade

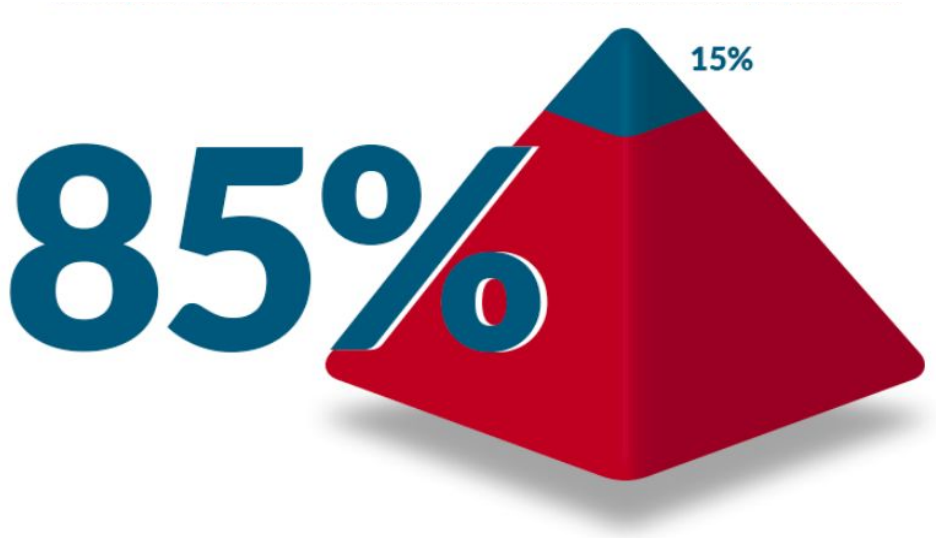
Liderança

Trab. em equipe

Autorganização

O que você acha que as empresas acham mais importante o profissional ter, Hard Skills ou Soft Skills ?

Uma pesquisa conduzida pela Harvard University, Carnegie Foundation e Stanford Research Center concluiu que 85% do sucesso no trabalho vem de ter habilidades sociais e pessoais bem desenvolvidas, e apenas 15% do sucesso no trabalho vem de habilidades e conhecimentos técnicos (hard skills).



Fonte

Dinâmica da aula

Visando não só o treinamento de Hard Skills como também o de Soft Skills a dinâmica da nossa aula será interativa, compostas de perguntas e respostas onde todos devem participar, dessa forma podemos treinar as Soft Skills (Habilidades pessoais) de **Comunicação**, **Liderança** e **Proatividade** além das Hard Skills (Habilidades técnicas) onde entenderemos diversos conceitos técnicos como **Como testar software**, **Plano de testes**, **Casos de testes** entre outros conhecimentos que vamos aprender no decorrer do curso.

O que é QA ?

QA, sigla inglesa para “**Quality Assurance**” que no português ficaria algo do tipo “**Garantia de Qualidade**”.

O que é qualidade?

O que é garantia?

QA, sigla inglesa para “**Quality Assurance**” que no português ficaria algo do tipo “**Garantia de Qualidade**”.

O que é qualidade?
A qualidade é extremamente difícil de definir e é simplesmente declarada: "Adequada para o uso ou finalidade." É tudo uma questão de atender às necessidades e expectativas dos clientes com relação à funcionalidade, design, confiabilidade, durabilidade e preço do produto.

O que é garantia?
A garantia nada mais é do que uma declaração positiva sobre um produto ou serviço, o que dá confiança. É a certeza de um produto ou serviço, que funcionará bem. Oferece a garantia de que o produto funcionará sem problemas de acordo com as expectativas ou requisitos.

QA no dia a dia

O QA deve, acima de tudo, garantir a qualidade ou seja, a necessidade em qualquer projeto de garantir que o software que estamos entregando ao cliente é exatamente o que eles querem. Entretanto, os Analistas de Qualidade são os pensadores que se especializam e se apropriam da qualidade do projeto. Isso é um desafio porque todos em uma equipe são responsáveis pela qualidade e, dependendo da composição da equipe, diferentes aspectos de qualidade podem ser tratadas por pessoas diferentes.

QAs se especializam na qualidade do software da sua equipe pois este deve ser muito envolvido no trabalho de um controle de qualidade. Na verdade, o controle de qualidade pode parecer como o trabalho num estilo camaleão, mudando suas responsabilidades diárias à medida que a equipe aprende e cresce, e o projeto passa por mudanças.

QA no contexto geral das empresas

Na maioria das empresas “QA” é o nome do cargo do profissional que atua com Qualidade e testes de software, o termo do cargo também pode ser chamado de:

- QA Tester
- QE (engenheiro de qualidade)
- Analista de QA
- Analista de testes
- Analista de qualidade
- Analista funcional
- Homologador
- Engenheiro de testes
- Arquiteto de testes

Garantia de qualidade no contexto de desenvolvimento de software

A Garantia de Qualidade em desenvolvimento de Software é definida como um procedimento para garantir a qualidade dos produtos ou serviços de software produzidos/desenvolvidos e depois fornecidos aos clientes por uma determinada organização. A garantia de qualidade se concentra em melhorar o processo de desenvolvimento de software e torná-lo eficiente e eficaz de acordo com os padrões de qualidade definidos para produtos de software. A garantia de qualidade é popularmente conhecida como “QA”

Processo de desenvolvimento tradicional de software

Este processo sugere uma abordagem sequencial e sistemática para o desenvolvimento de software, essa sequência sistemática consiste nos seguintes processos

Análise de requisitos: Nesta etapa o cliente ou stakeholders são ouvidos e suas necessidades são anotadas como requisitos do software, ou seja, o que esse sistema deve fazer

Elaboração do projeto: Nesta etapa é definido as metas e o escopo do projeto

Desenvolvimento: Nesta etapa é realizado o desenvolvimento em si, ou seja a codificação

Testes: Nesta etapa é realizado os testes de software

Implementação: Nesta etapa é onde o software é implementado em sim, após essa etapa é dada manutenção no projeto

Processo de desenvolvimento tradicional de software *em prática*

Objeto de estudo

O que são requisitos funcionais e não funcionais ?

Engenharia de Requisitos

Engloba um conjunto de tarefas a serem executadas para gerar como produto final uma documentação de requisitos. Tudo o que estiver contido nos documentos possibilitará que o software seja criado, atualizado e reparado sempre que necessário de acordo com o que foi inicialmente estipulado. Essa engenharia divide-se em 7 etapas principais:

1. Concepção

2. Elicitação

3. Elaboração

4. Negociação

5. Especificação

6. Validação

7. Gerenciamento

Requisitos funcionais (O que o sistema vai fazer)

É nos requisitos funcionais onde há a materialização de uma necessidade ou solicitação realizada por um software. Porém, vários Requisitos Funcionais podem ser realizados dentro de uma mesma funcionalidade. São variadas as funções e serviços que um sistema pode fornecer ao seu cliente, listamos abaixo algumas das inúmeras funções que os softwares podem executar:

- Incluir/Excluir/Alterar nome em uma tela de manutenção de funcionário
- Geração de relatório de determinado período de vendas
- Efetuar pagamentos de compra através de crédito ou débito
- Consulta e alterações de dados pessoais de clientes
- Emissão de relatórios de clientes ou vendas
- Consulta de saldo ou estoque

Requisitos não funcionais (Como o sistema vai fazer)

Os Requisitos não Funcionais não estão relacionados diretamente às funcionalidades de um sistema, Tratados geralmente como premissas e restrições técnicas de um projeto os requisitos não funcionais são praticamente todas as necessidades que não podem ser atendidas através de funcionalidades.

- O tamanho pode ser medido em bytes e número de Chip de RAM.
- A velocidade está ligada ao tempo de utilização da tela, ou transações processadas por segundos.
- A métrica da portabilidade é o número de sistema-alvo.
- A facilidade de uso pode ser medida pelo número de janelas ou o tempo de treino
- A confiabilidade tem ligação com o tempo médio que o sistema pode vir a falhar, a disponibilidade ou até mesmo a taxa de ocorrência de falhas.

Vamos praticar...

O que são Histórias/Estórias de usuário

Histórias/Estórias de usuário

A História/Estória de Usuário, ou User Story, é a técnica de descrever a necessidade do usuário do produto de forma simples e leve com a utilização de metodologia ágil a História/Estória de Usuário agregou outros métodos como o que surgiu do método XP (Extreme Programming) que foca mais na comunicação pessoal.

Podem ser idealizadas diversas História/Estória de usuário, na qual, cada estória precisa resolver um “problema”, conforme descrito abaixo.

O método busca resolver a fórmula: Como <*ator*>, quero/posso/preciso <*ação*> para <*justificativa*>

Ou seja, é a conciliação do Ator (quem vai receber o benefício?), a Ação (o que pode ser feito, o que quero que seja feito?) e a Justificativa (o motivo para fazer aquilo).

INVEST

O importante é que todas as estórias respondam a regra INVEST:

- **Independentes:** uma estória de usuário não deve depender da outra para acontecer;
- **Negociáveis:** não ser obrigatório uma estória ser finalizada para que outra aconteça, permitindo dessa forma maior flexibilidade de trabalho;
- **Valorosa:** o quanto ela traz para o objetivo global [valor];
- **Estimável:** ser possível mensurar [complexidade];
- **Small** (pequenas): que sejam entregues rapidamente, dentro de um período determinado (sprints);
- **Testável:** tem que ser possível ser testado para permitir feedback contínuo e entrar novamente no ciclo.

O que é Gherkin

Gherkin

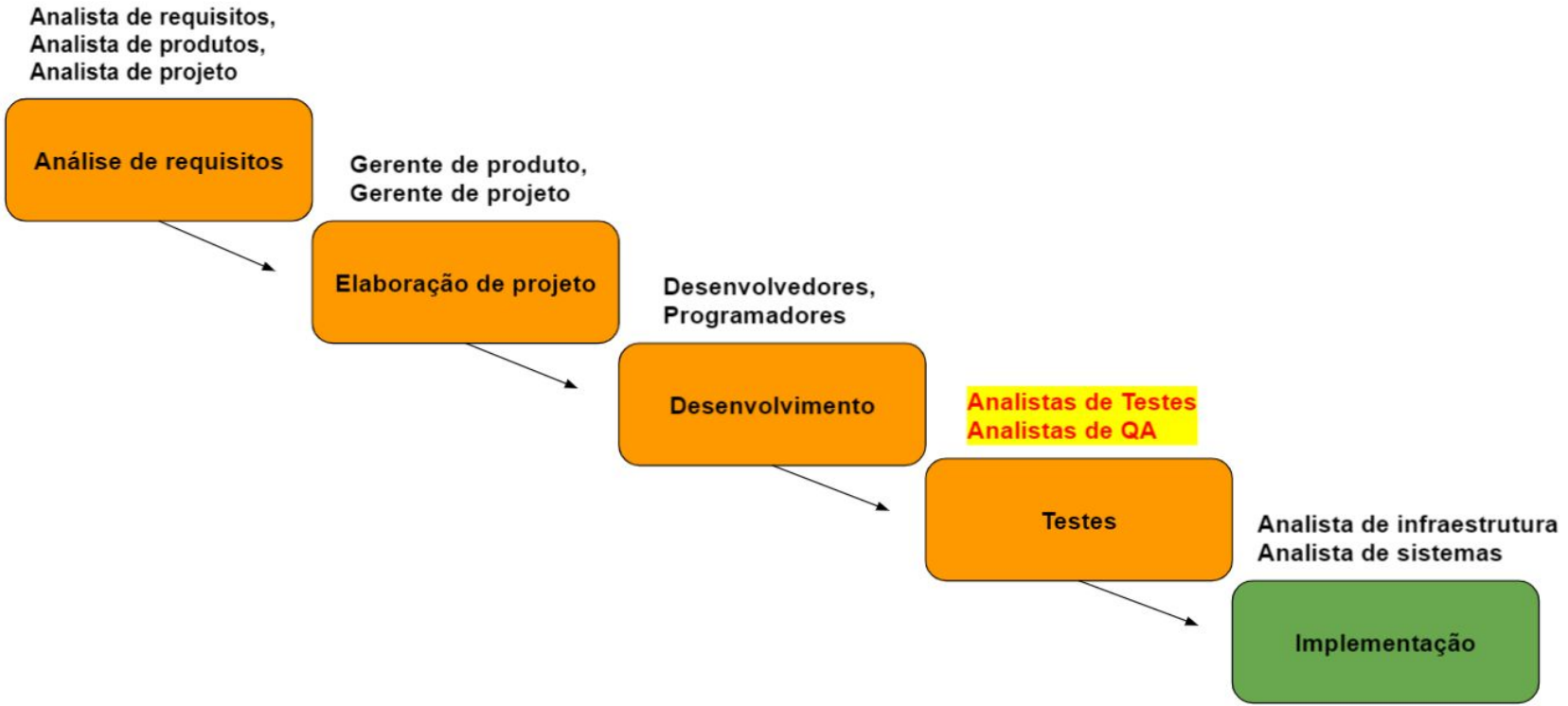
Assim como YAML e o Python, Gherkin é uma linguagem orientada a espaços, ela usa indentação para definir a estrutura. Os fins de linha encerram as declarações (denominados steps) e espaços ou tabs também podem ser usados para indentação (sendo que o indicado é você usar espaços para obter uma melhor portabilidade).

EX:

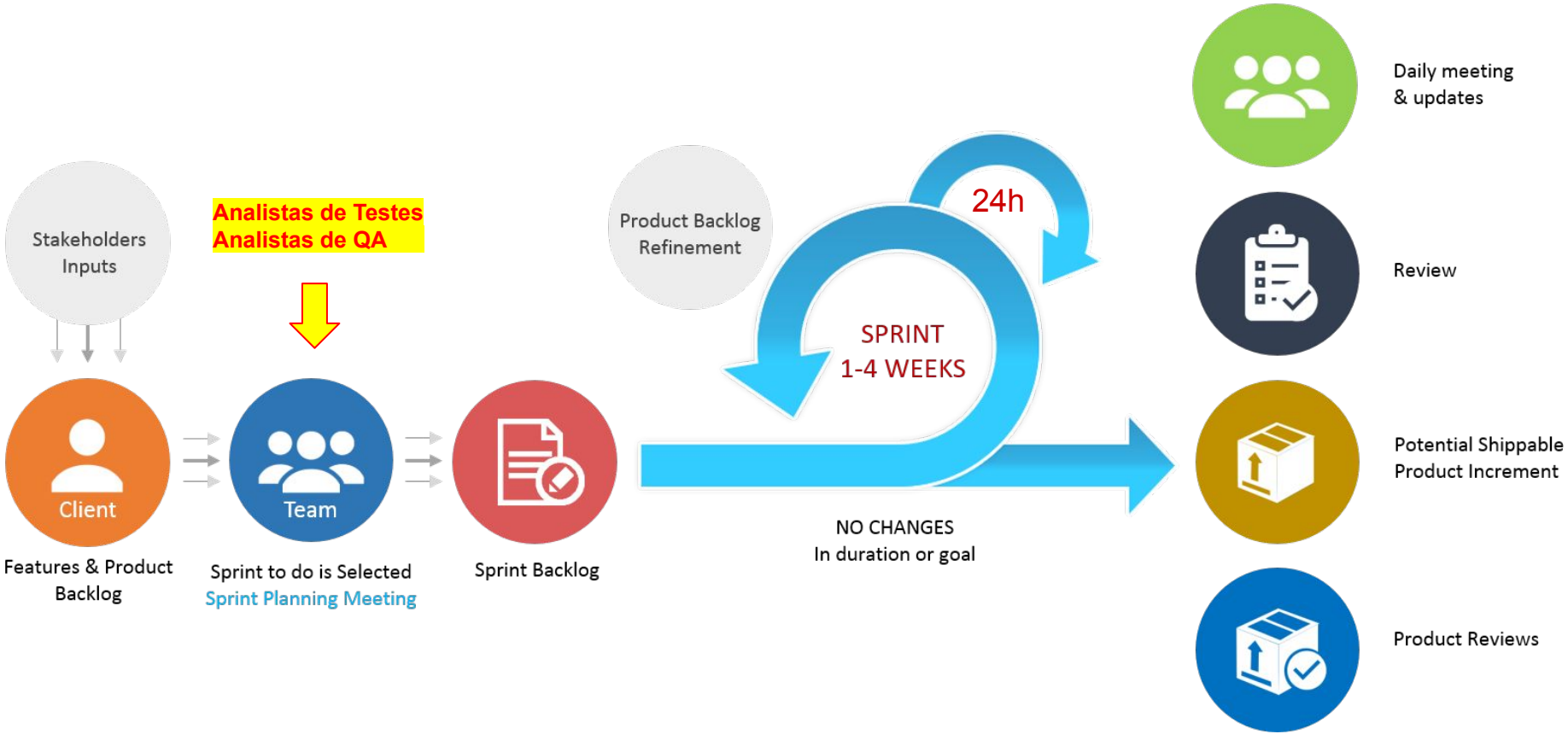
```
# language: pt
Funcionalidade: Login facebook
Cenário: Login com sucesso

Dado que um usuário está na tela de login
Quando preenche os campos de e-mail e senha de forma correta
E clicar no botão entrar
Então o usuario deve ter acesso ao seu facebook
```

Vamos praticar...

Modelo tradicional de desenvolvimento de software (Cascata)



Modelo Ágil de desenvolvimento de software (Framework Scrum)



Modelo Ágil (Framework Scrum)

O Scrum é uma estrutura que ajuda as equipes a trabalharem juntas. Semelhante a uma equipe de rugby (de onde vem o nome) treinando para o grande jogo, o Scrum estimula as equipes a aprenderem com as experiências, a se organizarem enquanto resolvem um problema e a refletirem sobre os êxitos e fracassos para melhorarem sempre.

Embora o Scrum sobre o qual estou falando seja mais usado pelas equipes de desenvolvimento de software, os princípios e as lições dessa estrutura podem ser aplicados a todos os tipos de trabalhos em equipe. Esse é um dos motivos de o Scrum ser tão popular. Muitas vezes considerado uma estrutura de gestão de projetos de agilidade, o Scrum descreve um conjunto de reuniões, ferramentas e cargos que atuam juntos para ajudar as equipes a organizarem e gerenciarem o trabalho.

Um time Scrum geralmente é formado por:

- **Product Owner (Dono do produto, responsável gerenciar as demandas)**
- **Scrum Master (Especialista em Scrum, responsável por implementar a cultura)**
- **Dev Team (Desenvolvedores, UX Designers e Analistas de Testes)**
- **Tech Lead e Especialista (Estes cargos não está no escopo do Scrum mas está presentes em muitos times e squads)**

Testador x Agile Tester

Os fundamentos do desenvolvimento ágil de software, *um testador em um projeto ágil trabalhará de forma diferente de um testador em um projeto tradicional*, os testadores devem compreender os valores e princípios que sustentam os projetos no formato ágil, e como farão parte integrante de uma abordagem de equipe junto com desenvolvedores e representantes de negócio. Os membros de um projeto ágil se comunicam com antecedência e com frequência, o que ajuda na remoção antecipada de defeitos e desenvolvimento de um produto de qualidade.

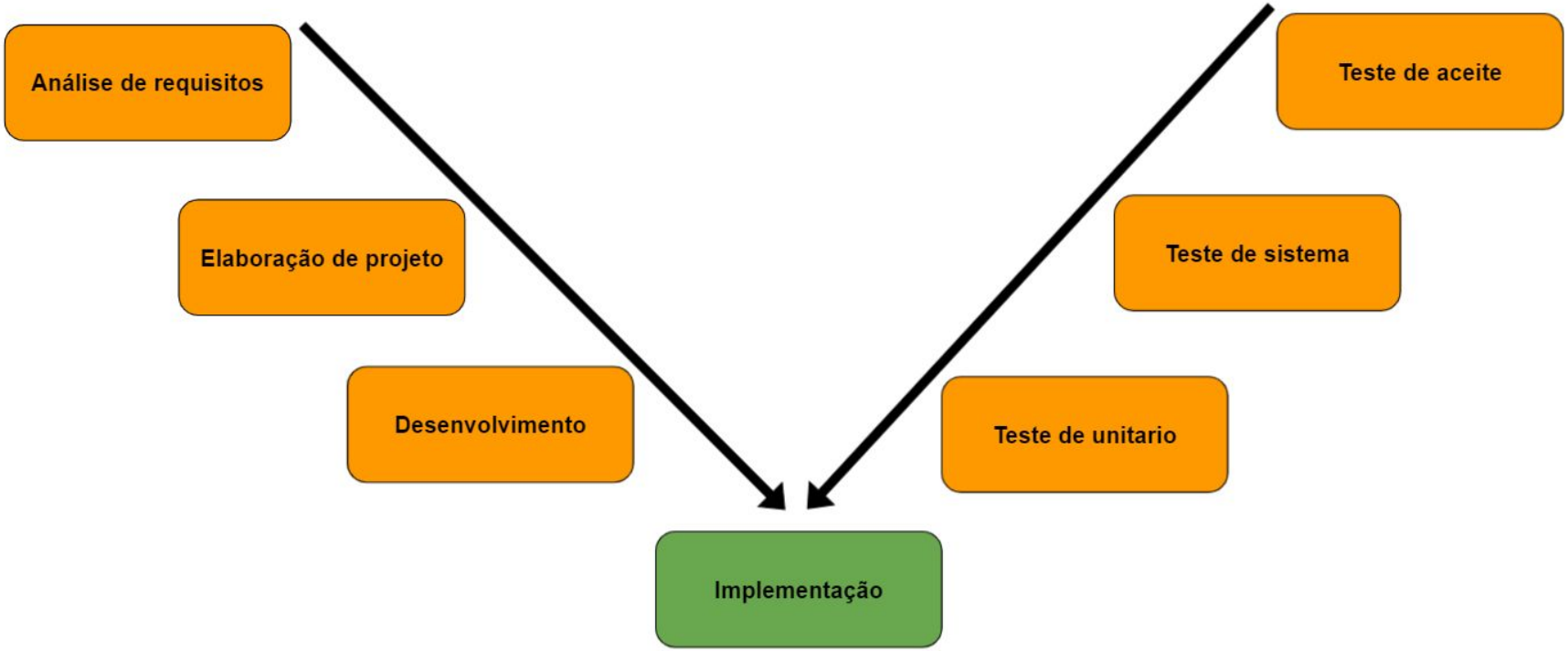
Ler material ISTQB

Qualidade ao longo do ciclo de vida do software (Cascata/Tradicional)

O ciclo de vida do software e as estratégias de qualidade e testes dependem do tipo de metodologias usadas no desenvolvimento do projeto, por exemplo na metodologia cascata, como estratégia mais conhecida de testes de software temos o “Modelo V” sendo este uma variação do modelo cascata, que demonstra como as atividades de testes estão relacionadas com análise e projeto.

Este modelo propõe que os testes de unidade e integração também podem ser utilizados para verificar o projeto de SW. Isto é, durante os testes de unidade e de integração, os programadores e a equipe de testes devem garantir que todos os aspectos do projeto foram implementados corretamente no código.

Modelo V (Cascata/Tradicional)



Qualidade ao longo do ciclo de vida do software (Ágil)

Uma das formas de garantir a qualidade no ágil seria através da metodologia de garantia de qualidade de ciclo definido denominado ciclo PDCA ou ciclo de Deming (Melhorias contínuas)

Plano - a organização deve planejar e estabelecer os objetivos relacionados ao processo e determinar os processos necessários para entregar um produto final de alta qualidade.

Do - Desenvolvimento e teste de Processos e também "fazer" mudanças nos processos

Verificar - Monitorar processos, modificar os processos e verificar se atende aos objetivos pré-determinados

Agir - Um testador de Garantia de Qualidade deve implementar ações que são necessárias para alcançar melhorias nos processos

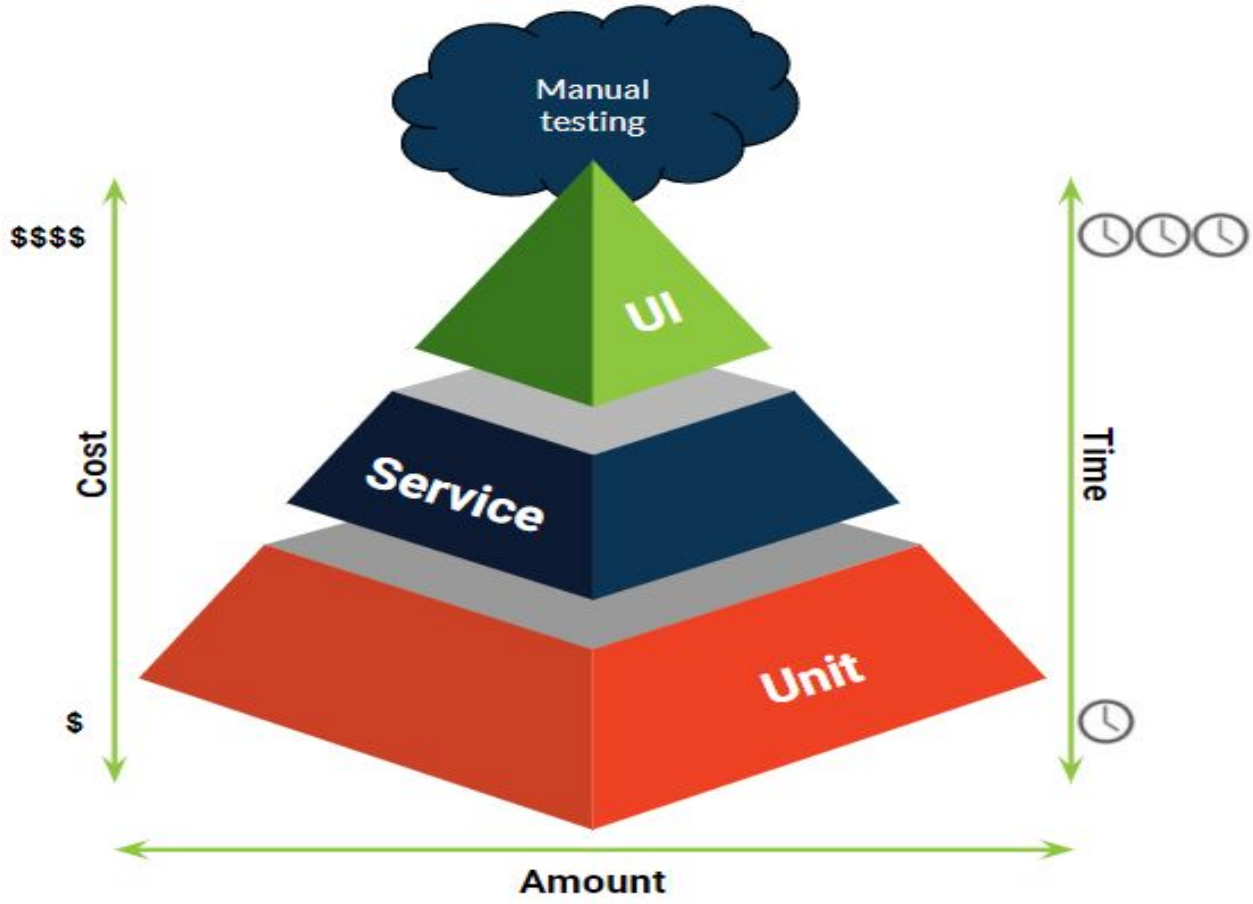

PLANEJAR
DESENVOLVER
CHECAR
AGIR
PDCA

Qualidade ao longo do ciclo de vida do software (Ágil)

Outra forma de garantir a qualidade de software é através da pirâmide de teste, existem diversas representações da pirâmide de teste, a que vou utilizar é a do Martin Fowler, A pirâmide de teste é uma maneira de pensar sobre como diferentes tipos de testes devem ser usados para criar uma estratégia de teste equilibrada.

Seu ponto essencial é que você deve ter muito mais testes unitários de baixo nível do que de alto nível executando através de uma GUI.

A pirâmide de teste aparece muito quando se fala em teste no Agile. Um problema comum é que as equipes confundem alguns conceitos como testes de ponta a ponta, testes de UI e testes de aceite.


Manual testing
UI
Service
Unit
$$$$
$
Cost
Time
Amount

O que é Mindset

O que é Mindset ?

De acordo com *Carol S. Dweck*, uma das maiores especialistas do mundo em psicologia social e psicologia do desenvolvimento, o sucesso, nas mais diferentes áreas da vida, não está exclusivamente vinculado a um talento ou habilidade especial. Na verdade, ele está, principalmente, relacionado ao resultado da maneira como encaramos a vida, ação também chamada de Mindset, que quer dizer, a atitude ou configuração mental que cada indivíduo tem, “Jeito de pensar”.

O termo Mindset significa “modelo mental”. Que nada mais é do que a maneira como uma pessoa pensa. É a configuração dos seus pensamentos. Assim, é a partir daqui que enfrentaremos as mais diversas situações do cotidiano. Também será através dele que seremos capazes de tomar decisões, trata-se da percepção que cada um tem da realidade em que está inserido, norteando sua vida.

BDD – Behavior Driven Development

Desenvolvimento orientado ao comportamento

BDD é uma técnica de desenvolvimento de software ágil que surge através de uma crítica de Dan North ao Test Driven Development(Desenvolvimento orientado a testes), onde ele visava otimizar o conceito de ‘verificação e validação’ já aplicado, e tornar mais eficiente a construção de cenários a serem testados e/ou desenvolvidos.

Para Kent Beck, criador do TDD, os testes devem ser escritos antes do código do software, assim irão falhar. Logo após, os desenvolvedores irão se basear nestes cenários falhos, irão implementar a aplicação de maneira a fazer os testes passar, e refatorar seu código até que fique mais limpo. De maneira cíclica. O que foi chamado de Red-Green- Refactor.

E está correto, porém a grande vantagem desta prática não é gerar testes, e sim pensar no design e nas regras negócios antes de escrever qualquer linha de código.

Teste de software, o que é ?

Teste de software
O que é ?

- “Pode ser usado para mostrar a presença de defeitos mas nunca para mostrar.” (Edsger W. Dijkstra)
- “Analisar um programa com a intenção de descobrir erros e defeitos.” (The Art of Software Testing, Glenford J. Myers)
- Investigação do software a fim de fornecer informações sobre sua qualidade em relação ao contexto
- Processo de utilizar o produto para revelar eventuais falhas
- Exercitar ou simular a operação de um programa ou sistema

Leitura técnica