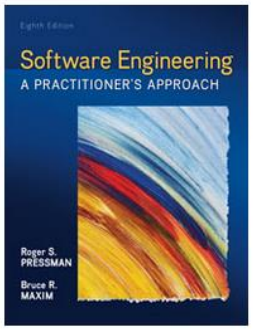
Unidade 4 - Princípios Fundamentais do Teste Ágil, Práticas e Processos

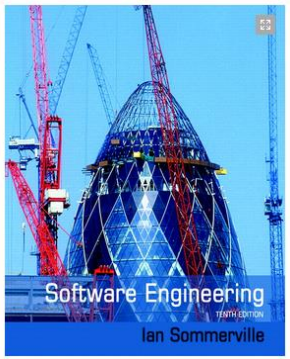
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecidovfreitas@gmail.com

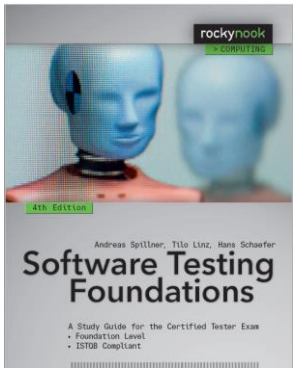
Bibliografia

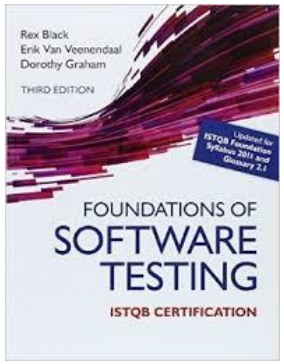
- **Software Engineering – A Practitioner's Approach – Roger S. Pressman – Eight Edition – 2014**
- **Software Engineering – Ian Sommerville – 10th edition - 2015**
- Engenharia de Software – Uma abordagem profissional – Roger Pressman - McGraw Hill, Sétima Edição - 2011
- Engenharia de Software – Ian Sommerville – Nona Edição – Addison Wesley, 2007
- Software Testing Foundations – Andreas Spillner , 2014 – 4th Edition
- Foundations of Software Testing ISTQB Certification – Rex Black, 2010
- **Agile Testing – Lisa Crispin, Janet Gregory, Mike Cohn Books, 2009**
- **Syllabus** – ISTQB – CTFL – AT – versão 2014BR

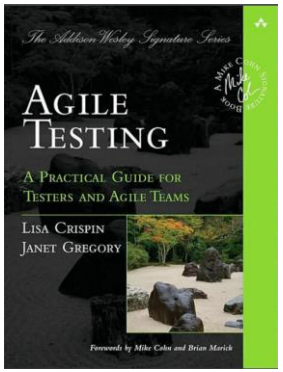


Software Engineering: A Practitioner's Approach, 8/e









Diferenças entre teste tradicional e teste ágil

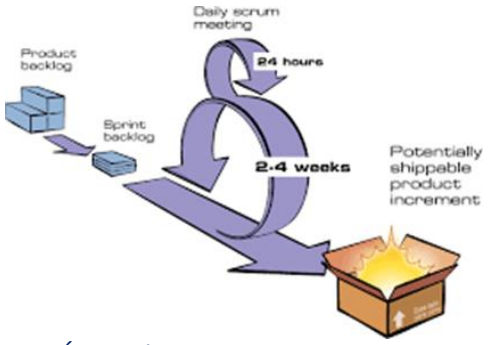
Syllabus 2.1

- Atividades de teste estão relacionadas com as atividades de desenvolvimento e, portanto, o teste varia em diferentes ciclos de vida;

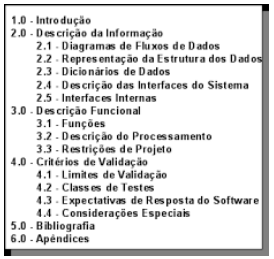
- **Testadores devem compreender as diferenças** entre os testes em modelos de ciclo de vida tradicional (sequencial – modelo V ou iterativo – RUP) e ciclos de vida ágil, a fim de trabalhar de forma eficaz e eficiente;

Atividades de Teste e Desenvolvimento

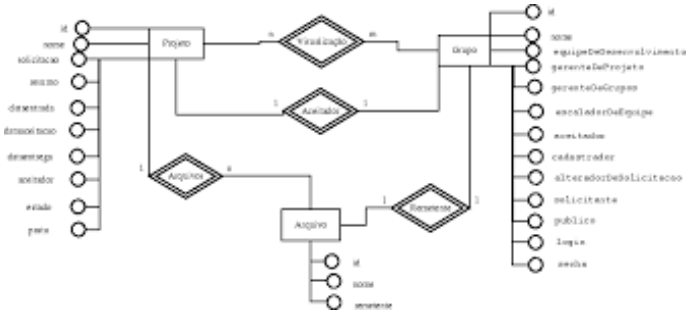
- Ciclo de vida tradicional pode demorar muitos meses, 1 ou 2 anos;
- Ciclos de vida de desenvolvimento ágil em geral, têm iterações curtas de 2 ou 4 semanas; Ao final de cada iteração entrega-se um **incremento** de software com **valor agregado ao cliente**;
- Em métodos ágeis, no início do projeto, há um período de **planejamento de lançamento,** seguido de uma sequência de iterações. No início de cada iteração, há um período de **planejamento de iteração**;
- Em processos ágeis, **atividades de teste ocorrem durante toda a iteração**, e não como uma atividade final.
- Um boa prática, em processos ágeis, é tratar **defeitos remanescentes** da iteração anterior, no início da próxima iteração, como parte do **backlog** para a iteração.



Produtos de Trabalho do Projeto



- Os produtos de trabalho do projeto de interesse para os testadores ágeis se enquadram em três categorias:
  - Produtos de trabalho orientados a **negócio**: descrevem o que é necessário (requisitos de especificações) e como usá-los (documentação do usuário, manuais);
  - Produtos de trabalho de desenvolvimento: descrevem como o Sistema é **construído** (diagramas ER, código, UML, testes de unidade automatizados);
  - Produtos de trabalho de teste: descrevem como o Sistema é **testado**: (estratégias de teste, planos de teste, etc)

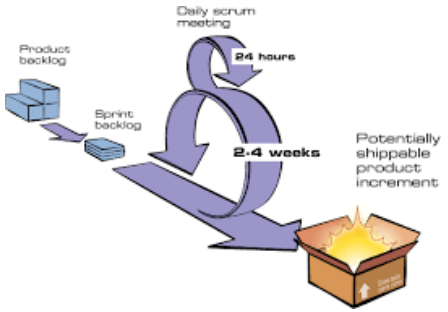


Níveis de Teste

- São atividades de teste que são logicamente relacionadas, muitas vezes, pela maturidade ou integridade do item em teste.

Níveis de Teste

- Durante uma iteração, qualquer estória de usuário geralmente progride **sequencialmente**, através das seguintes atividades de teste:
  - Teste de **unidade**, geralmente feito pelo desenvolvedor;
  - Teste de **aceitação da funcionalidade** que muitas vezes pode ser automatizado, podendo ser feitos por desenvolvedores ou testadores e envolve checagem dos critérios de aceitação da **estória do usuário**;
  - Teste de **validação da funcionalidade**, normalmente manual e pode envolver desenvolvedores, testadores e partes interessadas que trabalham de forma colaborativa para determinar se a funcionalidade está apta para uso (**feedbak real das parte interessadas**);



Níveis de Teste

- Testes de **regressão** consistem em reexecução dos testes para verificação da funcionalidade da iteração atual com as iterações anteriores (processo de integração contínua).

- Em alguns projetos ágeis pode haver um nível de teste do **Sistema**, podendo envolver a execução dos testes funcionais, bem como testes não-funcionais de desempenho, usabilidade, e outros tipos de teste pertinentes.

- Equipes ágeis podem empregar diversas formas de testes de **aceitação**. Podem ocorrer testes alfa internos, testes beta externos, testes de aceitação do usuário, testes de aceitação operacionais, testes de aceitação de regulamentação e testes de aceitação do contrato. Todos esses testes podem ocorrer, quer no final de cada iteração ou após uma série de iterações.

Gestão de Testes

- Projetos ágeis frequentemente envolvem o uso de **ferramentas automatizadas** para desenvolver, testar e gerenciar o desenvolvimento do software;
- Desenvolvedores geralmente usam ferramentas para análise estática (espécie de revisor de código) , testes unitários e de cobertura de código(áreas do código já cobertas e ainda não cobertas);
- Desenvolvedores verificam **continuamente** o código e os testes de unidade em um sistema de gerenciamento de configuração (**repositório com builds**, processo de integração contínua);
- Automação de testes, no entanto, não descarta a habilidade de um testador qualificado e experiente na detecção de defeitos (testes exploratórios).

Opções Organizacionais para Teste Independente

- Testadores independentes são, muitas vezes, mais **eficazes** na detecção de defeitos;
- Em algumas equipes ágeis, desenvolvedores criam muitos dos testes na forma de testes automatizados;
- No entanto, em equipes ágeis, dada a posição do testador há um risco de perda de independência e avaliação objetiva (**testador o tempo todo com o time**);
- Outras equipes ágeis mantêm equipes de teste totalmente independentes e separadas, e atribuem testadores sob demanda durante os dias finais de cada Sprint; No entanto, pressões de tempo, falta de compreensão das funcionalidades do produto e problemas de relacionamento, podem causar problemas com essa abordagem (**testador é acionado sob demanda, em cada Sprint**);
- Uma terceira alternativa, é ter uma equipe de teste separada e independente onde os testadores são designados para as equipes ágeis em uma base de longo prazo, no início do projeto, permitindo-lhes manter sua independência(**testador vinculado ao projeto**).

Opções Organizacionais para Teste Independente

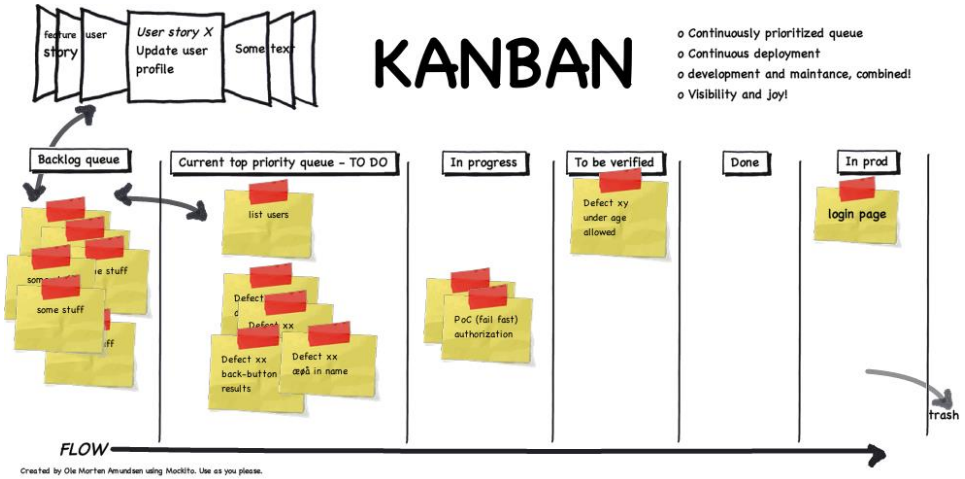
- A equipe de testes independente pode ter testadores especializados fora das equipes ágeis para trabalhar em atividades independentes de longo prazo, tais como o desenvolvimento de ferramentas de teste automatizado, ou testes de segurança, testes de performance, testes de usabilidade, testes de conformidade, testes de confiabilidade (software quebra?), testes de portabilidade, etc;

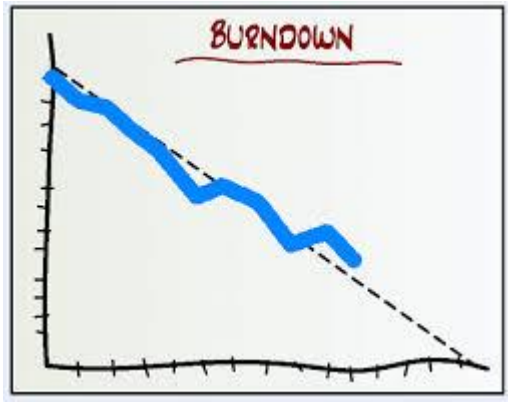
Status de Teste em Projeto Ágil

Syllabus 2.2

- Os testadores em equipes ágeis utilizam diversos métodos para registrar o progresso e o status dos testes. (mídia, portais, painéis wike, e-mails, bem como comunicação verbal durante reuniões);

- Um gráfico **Burndown** representa o volume de trabalho a ser realizado contra o tempo alocado para o lançamento ou iteração.





Progresso do Teste e Qualidade do Produto

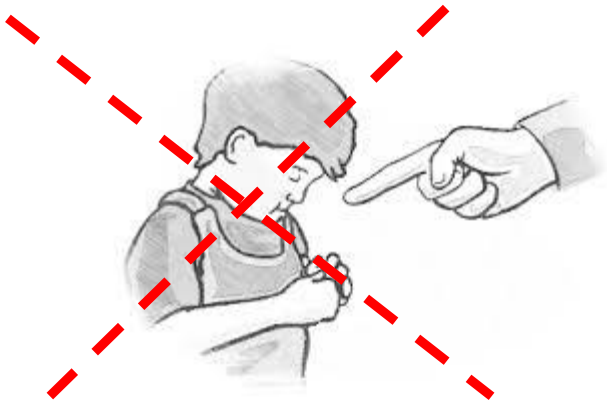
- Muitas equipes ágeis realizam **pesquisas de satisfação** do cliente para receber feedback sobre se o produto atende as expectativas dos clientes.

- Outras **métricas** podem ser utilizadas tais como: taxas de aprovação/reprovação de teste, taxas de detecção de defeitos, resultados de teste de confirmação e regressão;

- Pode-se também medir a densidade de defeitos, defeitos detectados, cobertura de requisitos, cobertura de código e rotatividade de código para melhorar a qualidade do produto.

Qualidade de Produto

- Em projetos ágeis, as métricas **não** devem ser utilizadas para premiar, punir ou afastar quaisquer membros da equipe;
- O projeto é desenvolvido por um **time**.



Gestão de Risco

- O **risco** de se introduzir uma regressão no desenvolvimento ágil é alto, devido ao código extenso (linhas de código adicionadas, modificadas ou apagadas de uma versão para outra);
- Visto que responder à mudança é um princípio chave ágil, é fundamental que as equipes invistam em automação de testes em todos os níveis o mais cedo possível;
- É fundamental que todos os ativos de teste (testes automatizados, casos de teste manuais e outros artefatos de teste) sejam **mantidos atualizados** com cada iteração (mantidos em algum lugar seguro e com controle de versão – alguma ferramenta de gestão de **configuração** de software);

Gestão de Risco de Regressão

- Como a **repetição** completa de todos os testes **raramente** é possível, testadores precisam alocar tempo em cada iteração para rever casos de teste manuais e automatizados de iterações anteriores e atuais para **selecionar** os casos de teste que podem ser candidatos ao teste de regressão;

- Testes escritos em iterações anteriores podem ter **pouco valor** em iterações posteriores devido a alterações de funcionalidades ou novas funcionalidades que alteram o comportamento de funções anteriores.

- Ao rever os casos de teste, os testadores devem considerar adequação para automação do máximo de testes possíveis de iterações anteriores e atuais. Esse esforço de teste de regressão reduzido **libera os testadores para testar mais a fundo novas funcionalidades e funções na iteração atual**.

Habilidades do Testador ágil

Syllabus 2.3

- Em equipes ágeis, testadores devem colaborar estreitamente com todos os outros membros da equipe e com as partes interessadas;
- Pessoas reagem diferentemente ao receberem a palavra "**bug**".

Testador

Desenvolvedor

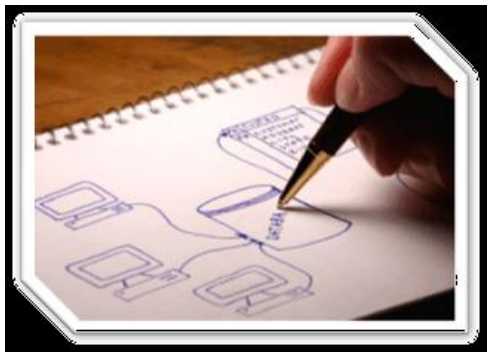
Gerente

Habilidades do Testador ágil

- Além das habilidades de um testador usual, o testador ágil deve ser competente em **automação de testes**, desenvolvimento orientado a testes, caixa branca, caixa preta e testes baseados em experiência;
- Como as metodologias ágeis dependem muito da colaboração, comunicação e interação entre os membros da equipe, bem como das partes interessadas fora da equipe, os testadores em equipes ágeis devem ter boas habilidades interpessoais;
- Devem ser **positivos** e orientados para solução com membros do time e partes interessadas;
- Devem mostrar pensamento crítico e cético orientada para **qualidade**, sobre o produto.

Habilidades do Testador ágil

- Ativamente **adquirir** informações das partes interessadas (ao invés de confiar inteiramente em especificações escritas);
- **Avaliar** com precisão e relatar os resultados dos testes, o progresso de teste e qualidade do produto;
- Trabalhar efetivamente para definir estórias de usuários **testáveis**, especialmente com os critérios de aceitação, com representantes dos clientes e partes interessadas (critérios objetivos e não subjetivos);
- **Colaborar** dentro da equipe, trabalhando em pares com os programadores e outros membros da equipe.

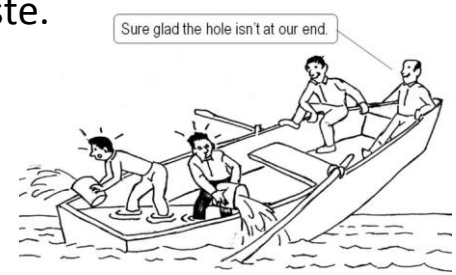


Habilidades do Testador ágil

- Responder à **mudança** rapidamente, incluindo alteração, adição ou melhora dos casos de teste;
- **Planejar** e **organizar** seu próprio trabalho
- **Crescimento contínuo** de **competências**, incluindo o crescimento de habilidades **interpessoais**, é essencial para todos os testadores, incluindo aqueles de equipes ágeis.

Funções de um Testador ágil

- A função de um testador em uma equipe ágil inclui atividades que geram e fornecem feedback, não só no status de teste, progresso de teste e qualidade do produto, mas também na qualidade do **processo (quality assurance)**;
- Compreender, implementar e atualizar a estratégia de teste (qual a melhor forma de testar o produto?);
- Medir e informar a cobertura do teste em todas as dimensões de cobertura aplicáveis;
- Garantir o uso adequado de ferramentas de teste;
- Participar ativamente das retrospectivas da equipe, sugerindo e implementando melhorias;
- Dentro de uma equipe ágil, cada membro da equipe é responsável pela qualidade do produto e desempenha um papel na execução das tarefas com o teste.



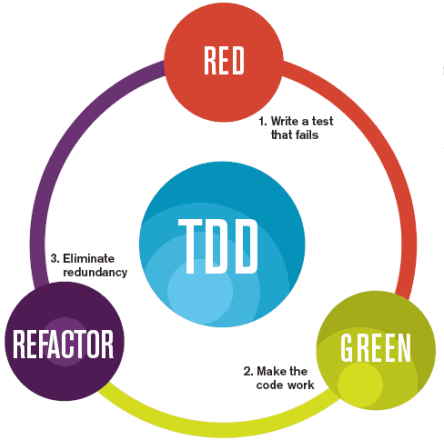
Métodos do teste ágil

Syllabus 3.1

- Existem três técnicas em uso entre as equipes ágeis para realizar testes entre vários níveis;
  - Desenvolvimento orientado a testes (**TDD**);
  - Desenvolvimento orientado para Teste de Aceitação (**ATDD**);
  - Desenvolvimento orientado para o Comportamento (**BDD**).

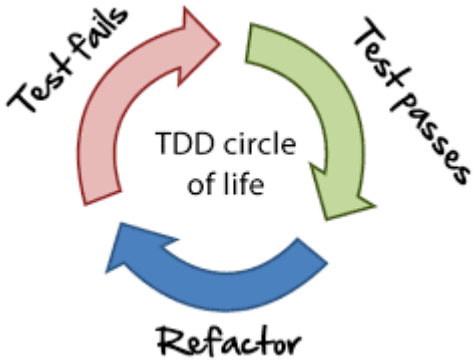
Desenvolvimento orientado a teste (TDD)

- O processo de desenvolvimento Orientado para Testes é:
  - **Adicionar** um teste que captura o conceito do programador do funcionamento desejado de uma pequena parte do código(teste **unitário** ou teste de **componente**);
  - **Realizar** o teste, o qual **falhará** uma vez que o código não existe;
  - **Escrever** o código e realizar o teste em um loop estreito **até o teste ser aprovado**.
  - **Refatorar** o código após a aprovação do teste, reexecutar o teste para garantir a continuidade o código refatorado;
  - **Repetir** esse processo para a próxima pequena parte do código, realizando os testes anteriores, bem como os testes adicionados;



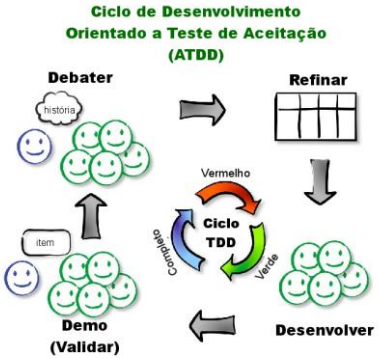
Desenvolvimento orientado a teste (TDD)

- Os testes escritos principalmente ao nível de **unidade** e são focados no código, embora os testes também possam ser escritos nos níveis de integração ou de sistema;
- Adquiriu popularidade através do **Extreme Programming** (**XP**), mas também é utilizado em outras metodologias ágeis e as vezes em ciclos de vida sequenciais;
- Ajuda os desenvolvedores a focar em resultados esperados claramente definidos;



Desenvolvimento orientado para teste de aceitação (ATDD)

- O desenvolvimento orientado para o teste de aceitação (**ATDD**) define critérios e testes de aceitação durante a criação de **estórias** de usuário;
- Abordagem **colaborativa** que permite que todas as partes interessadas compreendam como o componente de software tem que se comportar e o que os desenvolvedores, testadores e as partes interessadas precisam para garantir esse comportamento;
- Nesse teste predominam a linguagem do **negócio** (jargões do usuário).
- Permite a resolução rápida de defeitos e validação de comportamento da **funcionalidade**;
- Ajuda a determinar se os **critérios de aceitação** são cumpridos para a funcionalidade.

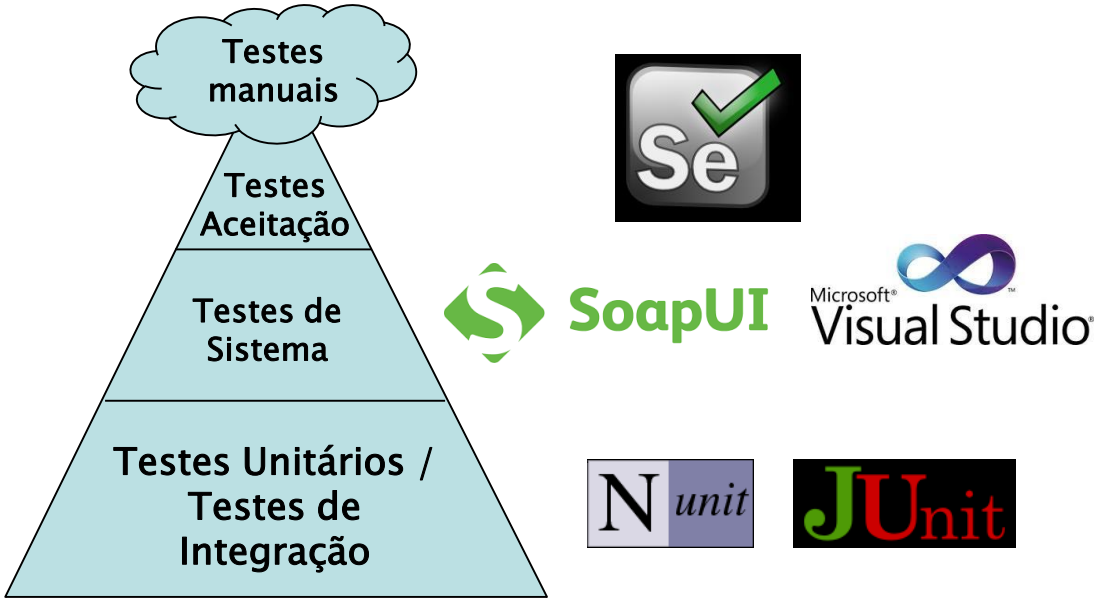


Desenvolvimento orientado para o Comportamento

- Permite que um desenvolvedor se concentre em testar o código com base no comportamento esperado do software;
- São geralmente mais fáceis de serem compreendidos por outros membros da equipe e partes interessadas;
- Critérios de aceitação são definidos com base na estrutura dado/quando/depois:
  - ❖ Dado algum contexto inicial, quando ocorre algum evento, em seguida assegurar alguns resultados.

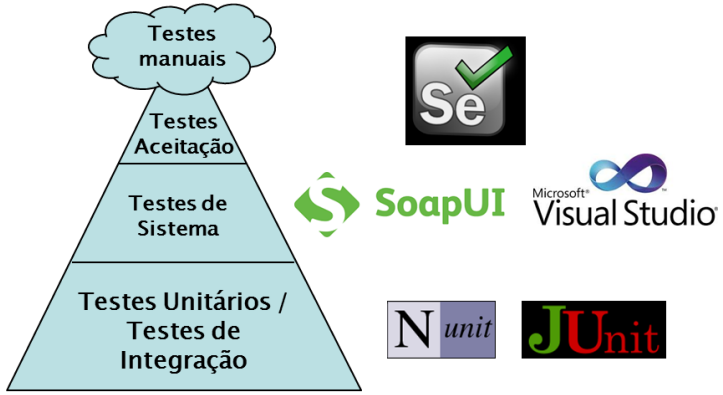
Pirâmide de Teste

- Um sistema de software pode ser testado em diferentes níveis;
- Níveis de teste típicos são, a partir da base da pirâmide para o topo, unidade, integração, Sistema e aceitação;
- A pirâmide de teste enfatiza um **grande número** de testes para os **níveis mais baixos** (base da pirâmide) e, conforme o desenvolvimento se move para **níveis superiores**, o **número de testes diminui** (topo da pirâmide);



Pirâmide de Teste

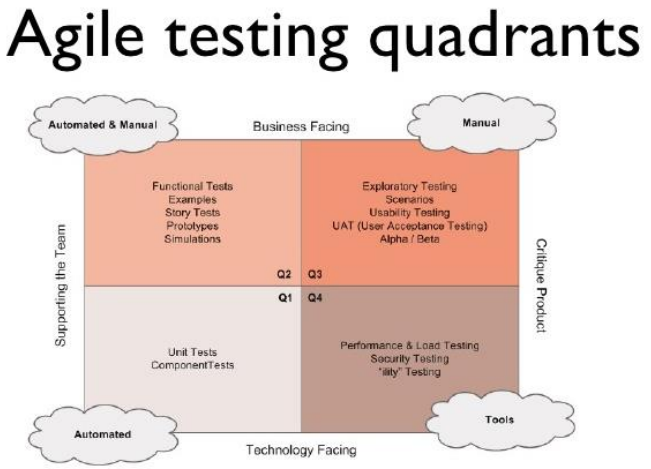
- Normalmente, os testes de unidade e nível de integração são automatizados e são criados usando ferramentas baseadas em API;
- Nos níveis de Sistema e de aceitação, os testes automatizados são criados usando ferramentas baseadas em GUI;
- O conceito de pirâmide de teste é baseado no princípio de controle de qualidade e testes (ou seja, a eliminação de defeitos o mais cedo possível no ciclo de vida).



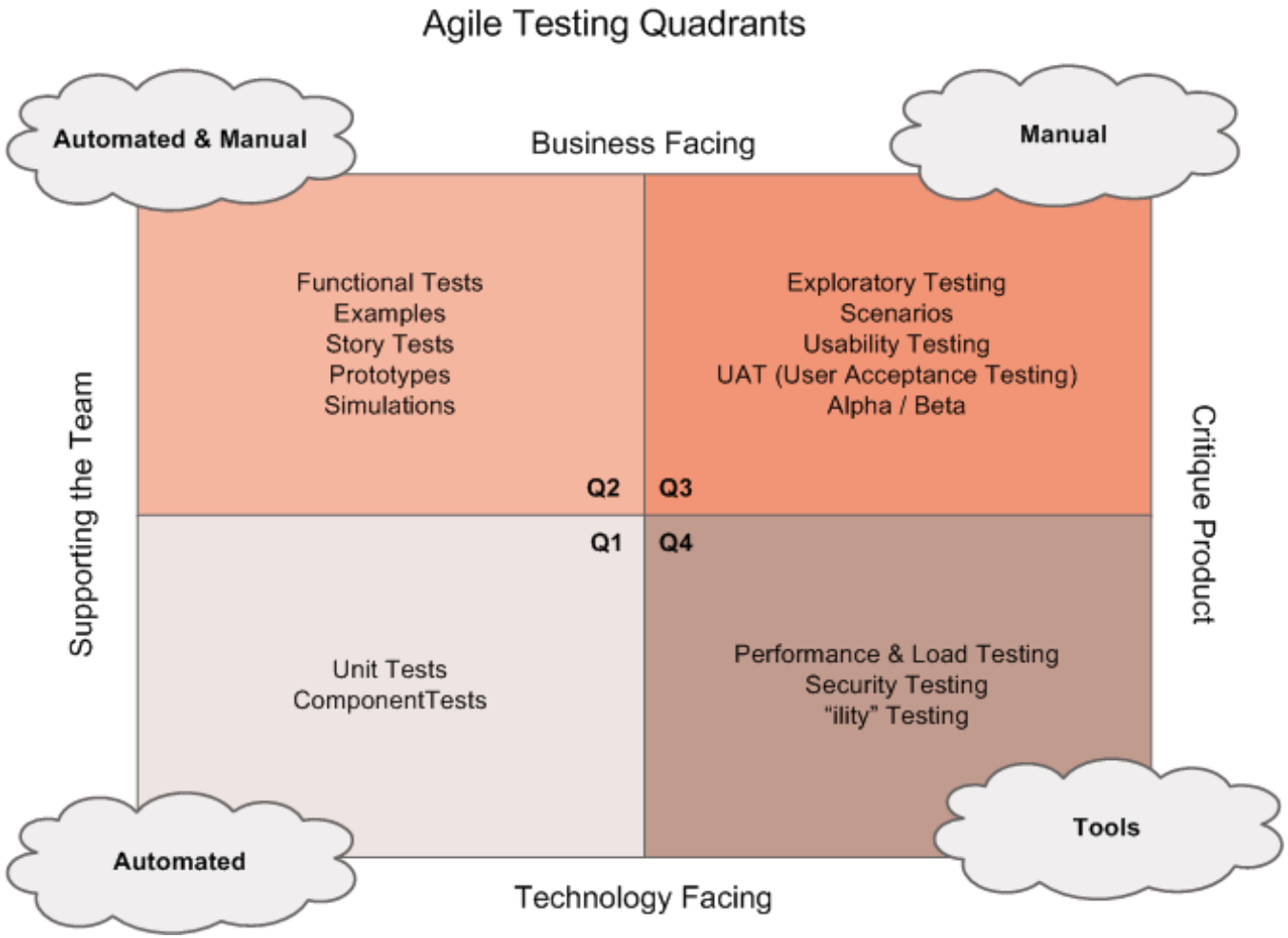
Quadrantes de Teste

- O modelo de quadrantes de teste e suas variantes ajuda a assegurar que todos os tipos de testes importantes e os níveis de teste sejam incluídos no ciclo de vida de desenvolvimento;

- Este modelo também fornece uma maneira de diferenciar e descrever os tipos de testes a todas as partes interessadas, incluindo desenvolvedores, testadores e representantes de negócio;

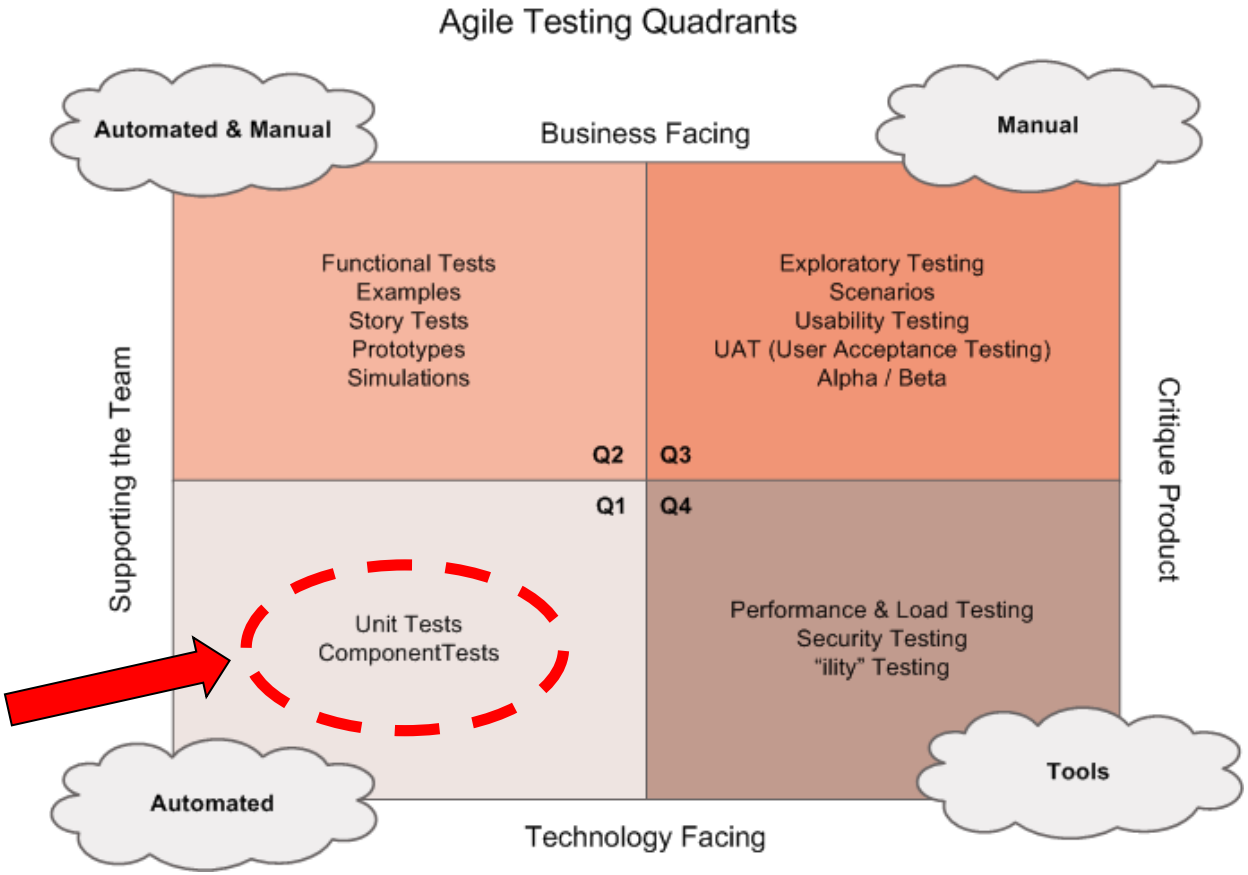


Quadrantes de Teste



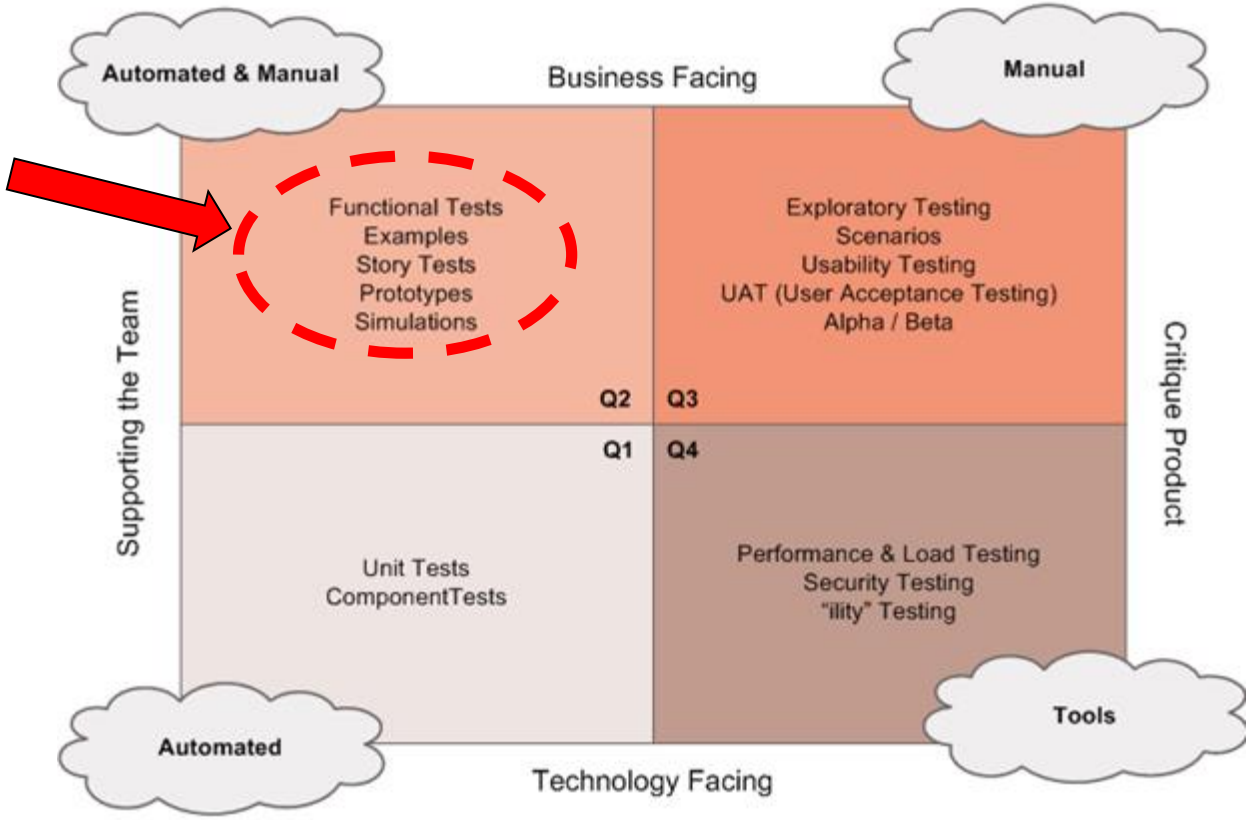
Quadrante Q1

- É o nível da unidade, voltado para **tecnologia** e apoia desenvolvedores;
- Este quadrante contém testes de **unidade**;
- Estes testes devem ser automatizados e incluídos no **processo de integração contínua**.



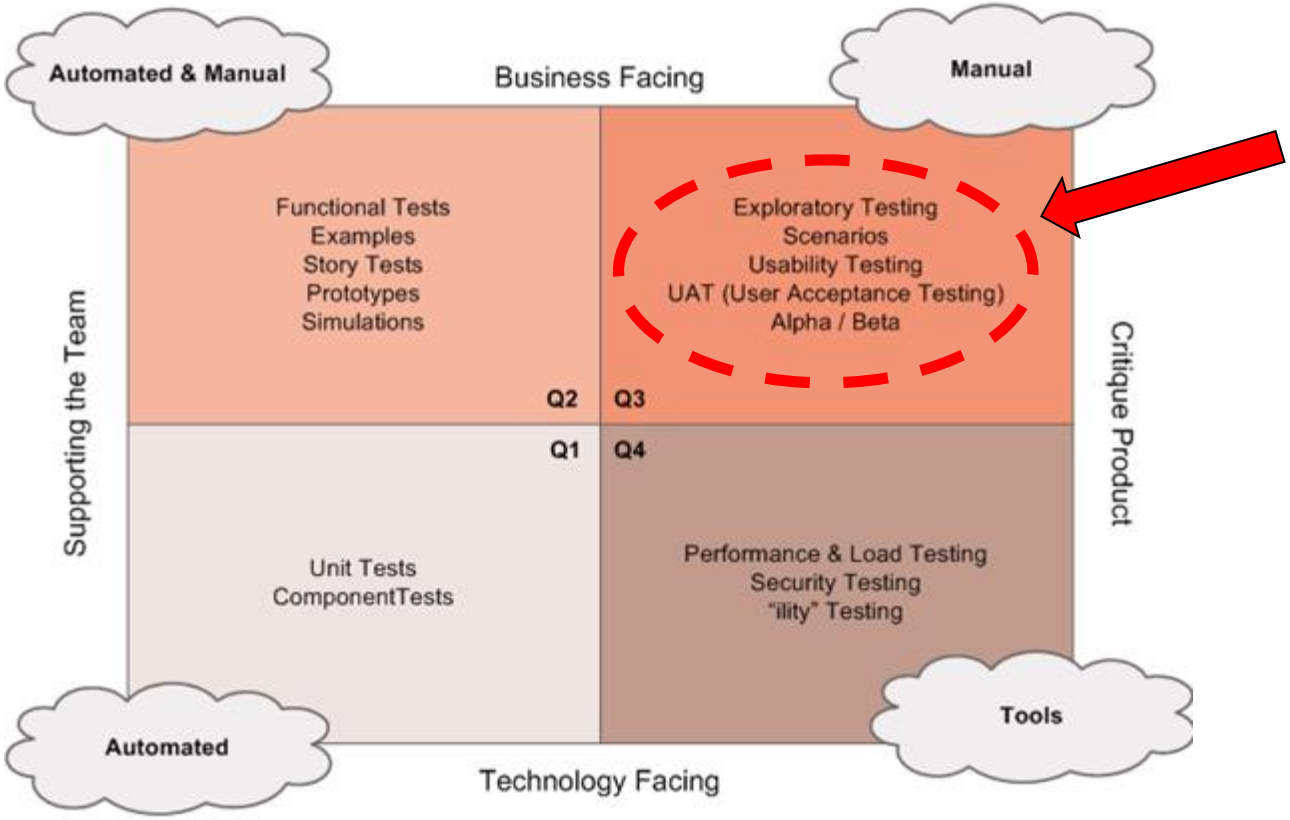
Quadrante Q2

- Voltado para **negócios** e confirma o comportamento do produto;
- Este quadrante contém testes funcionais, testes de estória, protótipos de experiência do usuário e simulações;
- Esses testes verificam os critérios de aceitação e podem ser manuais ou automatizados;
- São muitas vezes criados durante o desenvolvimento da estória do usuário e, assim, melhorar a qualidade das estórias.



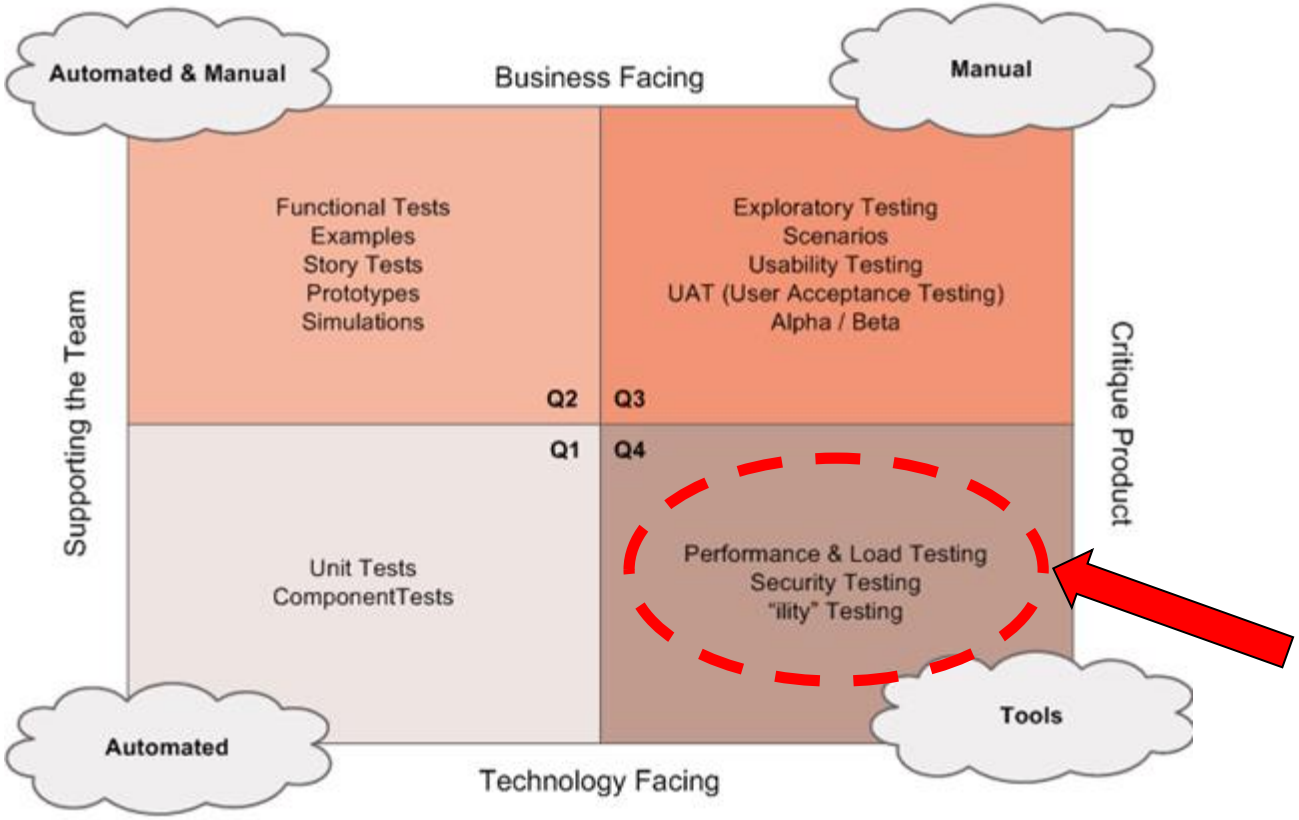
Quadrante Q3

- O quadrante Q3 é o nível de aceitação do sistema ou do usuário, voltado para o negócio, e contém testes que criticam o produto, utilizando cenários de dados realistas;
- Este quadrante contém testes exploratórios, cenários, fluxos de processos, testes de usabilidade, teste de aceitação do usuário, testes alfa e testes beta;
- Estes testes são muitas vezes manuais e orientados para o usuário.



Quadrante Q4

- O quadrante Q4 é o nível de **aceitação operacional** ou do sistema, orientado para tecnologia e contém testes que criticam o produto.
- Este quadrante contém desempenho, carga, estresse e testes de escalabilidade, testes de segurança, manutenção, gestão de memória, compatibilidade e interoperabilidade, migração de dados, infraestrutura e testes de recuperação;
- Estes testes são muitas vezes automatizados.



A função de um testador

- O testador é um importante membro de uma equipe ágil;
- Contribui com uma visão única para o projeto, pois possui capacidade tanto de vislumbrar questões de ordem técnica quando do negócio;
- Em muitos aspectos, o testador é uma pessoa que irá investigar o funcionamento do software e verificar se atende a todos os critérios de aceitação necessários para o cliente e para a equipe técnica;
- Trabalho em equipe é um princípio fundamental no desenvolvimento ágil.

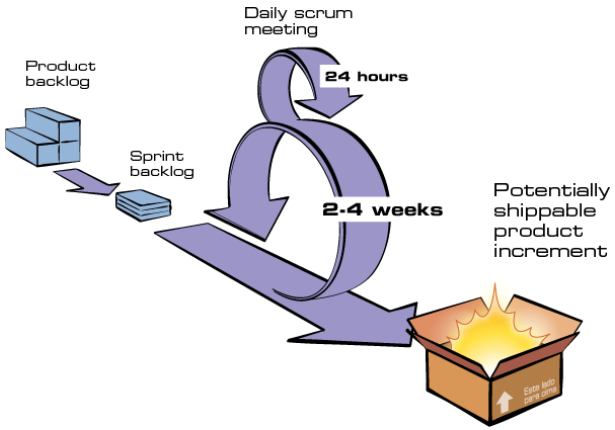
Característica das equipes ágeis

- **Multifuncional**: Cada membro da equipe traz um conjunto diferente de habilidades para a equipe. A equipe trabalha em conjunto. Ninguém toma decisões isoladamente. Assim, atividades do testador podem ser compartilhadas com outros membros do time;

- **Auto-organização**: A equipe pode consistir apenas de desenvolvedores, mas o ideal é que haja um ou mais testadores;

- **Co-localizado**: Os testadores se reúnem com os desenvolvedores e o proprietário do produto;

- **Colaborativo**: Os testadores colaboram com seus membros de equipe, outras equipes e partes interessadas, o proprietário do produto e o **Scrum Master**.



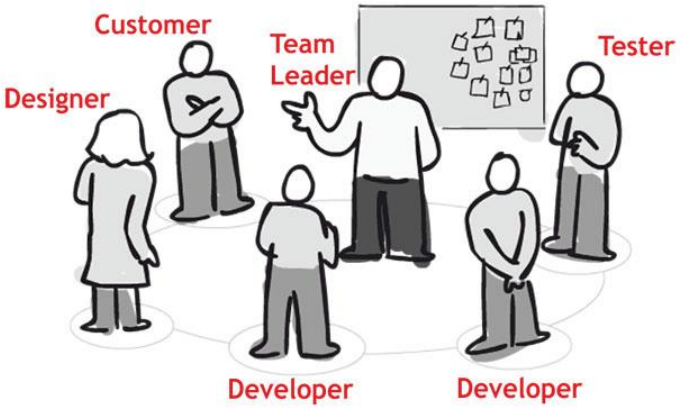
Característica das equipes ágeis

- **Capacitado**: As decisões técnicas de projeto e teste são tomadas pela equipe como um todo (desenvolvedores, testadores e Scrum Master), em colaboração com o proprietário do produto e outras equipes, se necessário;

- **Comprometido**: O testador tem o compromisso de questionar e avaliar o comportamento e as características do produto em relação às expectativas e necessidades do cliente.

Característica das equipes ágeis

- **Transparente**: Desenvolvimento e progresso de testes é visível no **quadro de tarefas**;
- **Credibilidade**: O testador deve garantir a credibilidade da estratégia de testes, sua implementação e execução, caso contrário, as partes interessadas não vão confiar nos resultados do teste. Isso é muitas vezes feito através do fornecimento de informações às partes interessadas sobre o processo de teste.



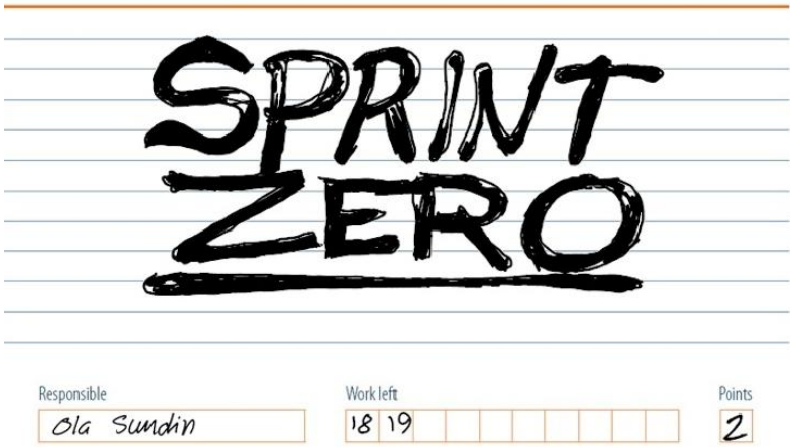
Característica das equipes ágeis

- **Aberto ao feedback**: O feedback é um aspecto importante para ser bem sucedido em qualquer projeto, especialmente em projetos ágeis. Retrospectivas permitem que as equipes aprendam com os sucessos e os fracassos;

- **Resiliente**: Os testes devem ser capazes de responder à mudança, como todas as outras atividades nos projetos ágeis.

Sprint Zero

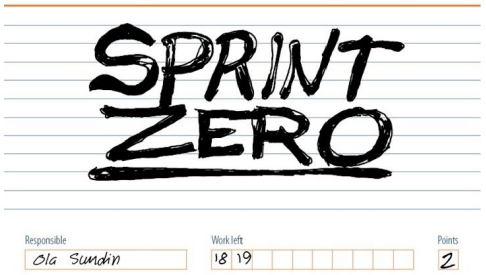
- Sprint zero é a primeira iteração do projeto, onde muitas atividades de preparação ocorrem;
- O testador colabora com a equipe nas seguintes atividades durante essa iteração;
  - Identificar o escopo do projeto (ou seja, o backlog do produto);
  - Criar uma arquitetura inicial do sistema e protótipos de alto nível.



Sprint Zero

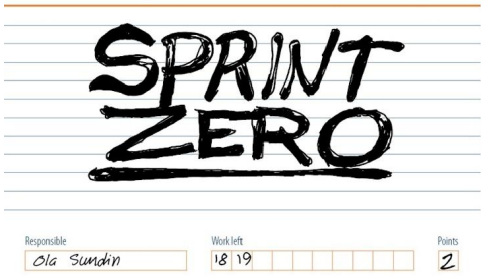
- Consiste em se **planejar**, adquirir e instalar as ferramentas necessárias (por exemplo, para gerenciamento de testes, gerenciamento de defeitos, automação de testes e integração contínua);

- Consiste em se criar uma **estratégia** de teste inicial para todos os níveis de teste, abordando (entre outros tópicos) escopo de teste, riscos técnicos , tipos de teste e metas de cobertura;



Sprint Zero

- Realizar uma análise de risco inicial de qualidade;
- Definir métricas de teste para medir o processo de teste, o progresso dos testes no projeto, e a qualidade do produto;
- Especificar a definição de “realizado”;
- Criar o quadro de tarefas;
- Definir quando continuar ou interromper o teste antes de entregar o sistema para o cliente.



Práticas de Teste ágil

- **Empilhamento**: Dois membros da equipe (por exemplo, um testador e um desenvolvedor, dois testadores, ou um testador e um proprietário do produto) se reúnem em uma estação de trabalho para realizar um teste ou outra tarefa Sprint;

- **Projeto de Teste Incremental**: Os casos de teste são gradualmente desenvolvidos de estórias de usuários e outras bases de testes, começando com testes simples e passando para testes mais complexos;

- **Mapeamento Mental**: É uma ferramenta útil no teste. Por exemplo, os testadores podem usar o mapeamento mental para identificar quais sessões de teste realizar, para mostrar estratégias de teste, e para descrever os dados de teste.

FreeMind
Mind Mapping Software

Avaliação de Riscos de Qualidade e Estimativa de Esforço de Teste

Syllabus 3.2

- Testadores em projetos ágeis podem usar as mesmas técnicas utilizadas em projetos tradicionais para **identificar** riscos de qualidade (ou riscos de produto), **avaliar** o nível de risco associado, **estimar** o esforço necessário para reduzir suficientemente esses riscos, e depois **mitigar** esses riscos através de projeto de teste, implementação e execução.

Avaliação de Riscos de Qualidade

- Um dos muitos **desafios** em testes é a **seleção**, **alocação** e **priorização** adequadas das condições de teste;

- Isso inclui determinar o volume adequado de **esforço** para alocar a fim de cobrir cada **condição** com testes, e sequenciar as provas resultantes de modo a otimizar a eficácia e eficiência do trabalho de teste a ser feito.

Avaliação de Riscos de Qualidade

- A **identificação** dos riscos, **análise** e estratégias de **mitigação** de risco podem ser utilizadas pelos testadores nas equipes ágeis para ajudar a determinar um número **aceitável** de **casos** de **teste** a serem realizados;
- **Risco** é a possibilidade de um resultado ou evento negativo ou indesejável;
- O **nível de risco** é detectado através da avaliação da **probabilidade de ocorrência do risco** e do **impacto do risco**.

Exemplos de Riscos de Qualidade

- **Cálculos** incorretos em relatórios (um risco funcional relacionado com a **acurácia**);
- Resposta **lenta** à entrada do usuário (um risco não funcional relacionados com a **eficiência** do tempo de resposta);
- Dificuldade na **compreensão** de telas e campos (um risco não funcional relacionado com a **usabilidade** e **inteligibilidade**).

Avaliação de Riscos de Qualidade

- Uma iteração começa com o planejamento da iteração, que culmina em um quadro de tarefas estimadas em um quadro de tarefas;
- Estas tarefas podem ser **priorizadas** em parte com base no nível de risco de qualidade associado às mesmas;
- Tarefas associadas a **riscos mais elevados** devem começar **mais cedo** e envolvem mais esforço de teste;
- Tarefas associadas a **riscos menores** devem começar **mais tarde** e envolvem menos esforço de teste.

Exemplo de Processo de Análise de Risco

1. Reunir os membros da equipe ágil, incluindo o testador ou testadores;

2. Listar todos os itens do backlog da iteração atual (por exemplo, em um quadro de tarefas);

3. Identificar os riscos de qualidade associados a cada item, considerando-se todas as características relevantes de qualidade;

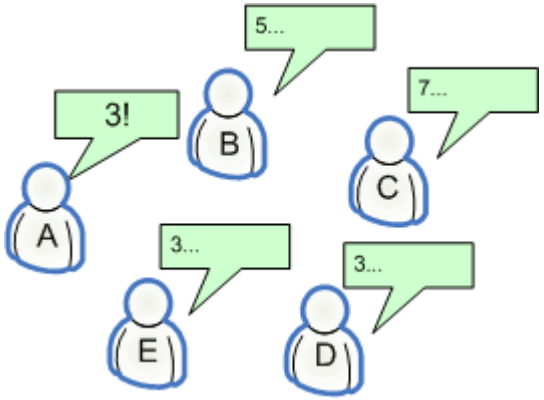
4. Avaliar cada risco identificado, que inclui duas atividades: categorizar o risco e determinar o seu nível de risco com base no impacto e na probabilidade de defeitos;

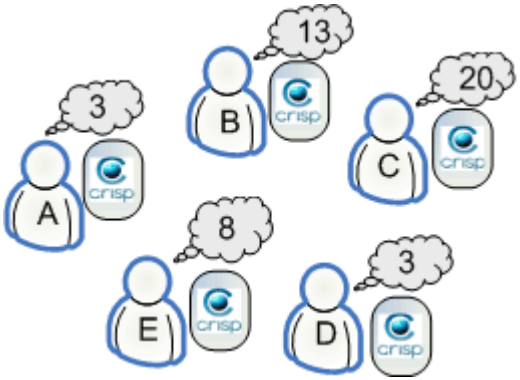
5. Determinar a extensão do teste proporcional ao nível de risco;

6. Escolher a técnica de teste apropriada para mitigar cada risco, com base no risco, nível de risco, e a característica de qualidade pertinente.

Estimativa do esforço de Teste

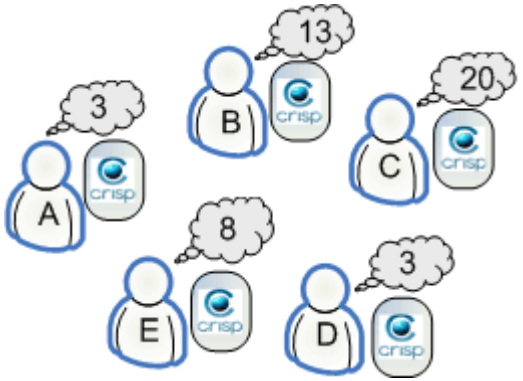
- Durante o planejamento de lançamento, a equipe ágil estima o esforço necessário para completar o lançamento;
- A estimativa aborda também o **esforço de teste**;
- A técnica de estimativa comum utilizada nos projetos ágeis é o **pôquer do planejamento**, uma técnica baseada no **consenso**.





Planning Poker

- O proprietário do produto ou cliente lê uma estória de usuário para os avaliadores;
- Cada avaliador tem um baralho de cartas com valores semelhantes para a sequência de Fibonacci (ou seja, 1,1,2,3,5,8,13,21,34,55,89,...)ou qualquer outra progressão de escolha (por exemplo, os tamanhos da camisa que variam de extra-pequeno a extra-grande.
- Os valores representam o número de pontos da estória, dias de esforço, ou outras unidades nas quais a equipe estima.



Planning Poker

- A sequência de Fibonacci é **recomendada**, pois os números na sequência refletem que a incerteza cresce proporcionalmente com o tamanho da estória;

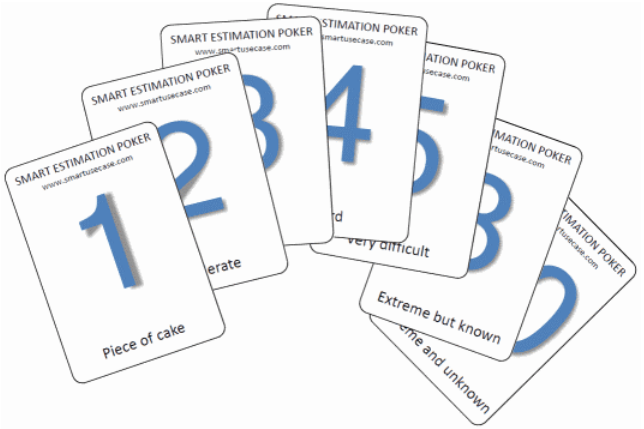
- A estimativa **alta** significa geralmente que a estória **não** é bem compreendida ou deve ser dividida em várias estórias menores.

Estimativa do Esforço de Teste

- Os estimadores discutem a funcionalidade e fazem perguntas sobre o proprietário do produto, conforme necessário;
- Aspectos como desenvolvimento e esforço de teste, complexidade da estória e escopo dos testes desempenham um papel fundamental na estimativa.

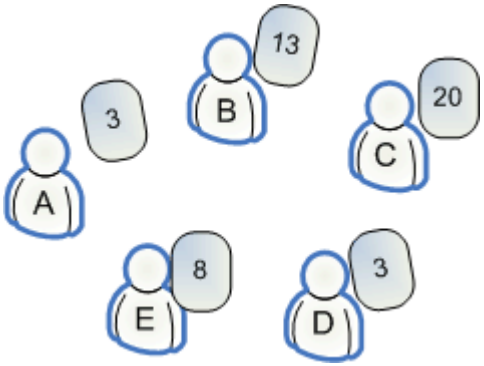
Planning Poker

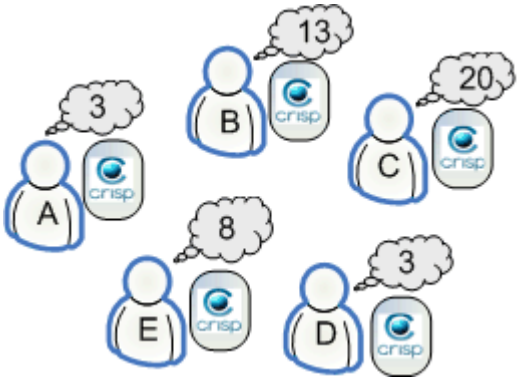
- Quando a funcionalidade é amplamente discutida, cada estimador seleciona privativamente uma carta para representar a sua estimativa;
- Todos os cartões são então revelados ao mesmo tempo;
- Se todos os estimadores selecionaram o mesmo valor, este se torna a **estimativa**;



Planning Poker

- Caso contrário, os estimadores discutem as diferenças nas estimativas depois que a rodada de pôquer é repetida, até que seja alcançado um **acordo**, seja por **consenso** ou pela aplicação de **regras** (por exemplo, usar a mediana, usar a maior pontuação) para limitar o número de rodadas do pôquer.





Estimativa do Esforço de Teste

- Essas discussões garantem uma estimativa confiável do esforço necessário para completar itens de backlog do produto solicitados pelo proprietário do produto e ajudam a melhorar o conhecimento coletivo do que tem que ser feito.