



# WACAD018 – Processo de Desenvolvimento Ágil - Aula 01

---

Júlia Luiza

[jlslc@icomp.ufam.edu.br](mailto:jlslc@icomp.ufam.edu.br)

# Cronograma: Aula 01

---

- **Breve Histórico do desenvolvimento ágil**
- **Manifesto Ágil:**
  - Valores, Princípios, Consequências.
- **Métodos Ágeis:**
  - Onde aplicar, Vantagens.
- **Conceitos importantes em ambientes ágeis**
  - CI/CD, Squad, Task, Board;
  - User stories;
  - Backlog de Produto;
- **Prática 01: User Stories no board Jira**
- **SCRUM Parte 1:**
  - O que é, Princípios.

# Avaliação

---

## Trabalhos práticos incrementais em grupo:

- Aula 1 -> Prática 1 (P1)
- Aula 2 -> Prática 2 (P2)
- Aula 3 -> Prática 3 (P3)

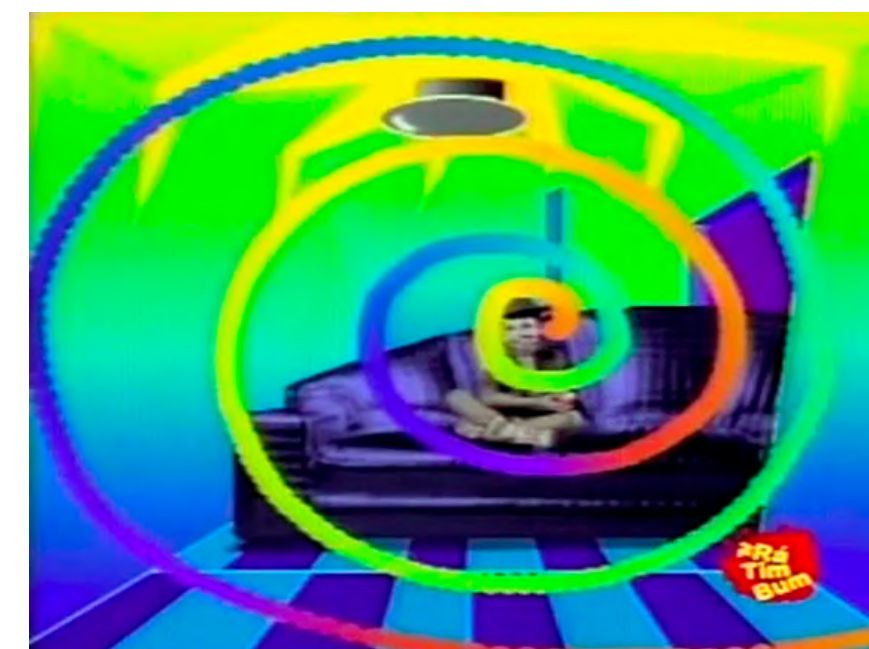
$$\text{Média disciplina} = (P1 + P2 + P3) / 3$$



# Breve histórico

Em meados da década de 80 e início da década de 90, havia um senso comum de que a melhor forma de desenvolver software era por meio de um **planejamento cuidadoso, com formalidades e documentação extensa**.

- O que, com o tempo, acabou causando uma insatisfação geral quanto ao ***overhead*** - sobrecarga - advindo dos métodos de projeto de software da época;
- Era **comum um atraso de vários anos** entre a validação de uma necessidade de negócios pela empresa e a entrega de uma aplicação funcional para supri-la;
- Por isso, **no final da década de 90, um grupo de especialistas se reuniu para discutir maneiras de aprimorar o desenvolvimento de software** e outras alternativas aos processos da época, como por exemplo o modelo cascata.

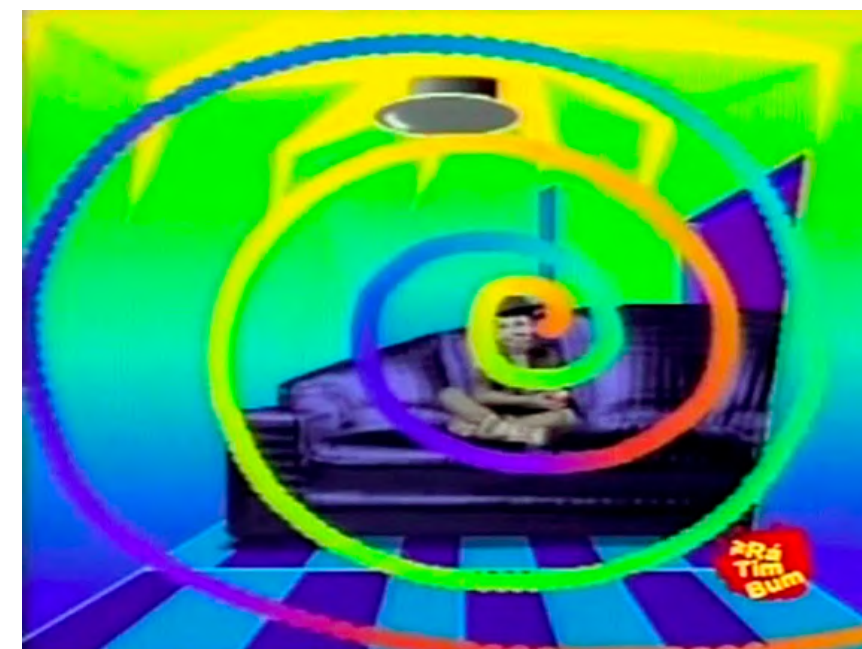


# Breve histórico

---

A decorrer da reunião, um consenso comum sobre aspectos importantes em desenvolvimento de software foi se formando entre os especialistas.

- A partir daí, decidiram escrever um **documento em formato de manifesto**, compilando as ideias disruptivas para a época;
- Após considerar vários nomes, decidiram que a palavra “**ágil**” melhor captava a abordagem proposta;
- Dessa forma, surgiu o documento que desencadaria o "**Agile Manifesto**", ou "**Manifesto Ágil**";
- No documento, estavam contidos conceitos sobre o que o manifesto defendia em relação ao **desenvolvimento de software**, **seus princípios e valores**.



# Manifesto Ágil

---

Publicado em 2001, o Manifesto Ágil trouxe os princípios fundamentais da metodologia ágil para desenvolvimento de software.

- "Os signatários do manifesto acreditavam que a **colaboração próxima entre os membros da equipe, a resposta rápida às mudanças e a entrega de valor incremental** eram essenciais para o sucesso de um projeto.";
- No documento, descrevem as **quatro características** que eles acreditam ser mais importantes do que outras questões;
- E também **12 princípios** principais;
- Contudo, o manifesto Ágil **não tem como objetivo prescrever um conjunto de procedimentos**. Ele é um guia para uma nova maneira de pensar sobre o desenvolvimento de aplicações.



# Manifesto Ágil: Valores

---

## **Manifesto para Desenvolvimento Ágil de Software**

Estamos descobrindo maneiras melhores de desenvolver software, fazendo-o nós mesmos e ajudando outros a fazerem o mesmo. Através deste trabalho, passamos a valorizar:

**Indivíduos e interações** mais que processos e ferramentas  
**Software em funcionamento** mais que documentação abrangente  
**Colaboração com o cliente** mais que negociação de contratos  
**Responder a mudanças** mais que seguir um plano

Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

# Manifesto Ágil: Valores

---

- **Indivíduos e interações mais** que processos e ferramentas
  - Processos e ferramentas são importantes, mas devem ser simples e úteis. No final das contas, o desenvolvimento de software é uma atividade humana, portanto a qualidade da interação entre as pessoas pode resolver boa parte dos problemas do dia a dia.
- **Software em funcionamento mais** que documentação abrangente
  - Como dito no próprio manifesto, a documentação não é desmotivada, mas é indicado que seja reduzida para conter somente o necessário. Antes disso, o que mais agrega valor é de fato uma entrega, ou seja, o software em funcionamento.





# Manifesto Ágil: Valores

---

- **Colaboração com o cliente mais** que negociação de contratos
  - Em vez de focar exclusivamente em acordos contratuais rígidos, o objetivo é ter uma parceria colaborativa na qual o cliente e a equipe de desenvolvimento trabalham juntos para atingir os objetivos comuns.
- **Responder a mudanças mais** que seguir um plano
  - Ao invés de seguir um plano inicial de forma inflexível, o foco está em ser ágil e capaz de responder rapidamente a novas necessidades, oportunidades ou desafios que surgem durante o processo de desenvolvimento.



# Manifesto Ágil: Princípios

---

## Princípios por trás do Manifesto Ágil

*Nós seguimos estes princípios:*

Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.

Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento.

Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.

Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.

# Manifesto Ágil: Princípios 1-6

---

1. Satisfação do cliente através da **entrega contínua e adiantada** de software com valor.
2. **Mudanças nos requisitos são bem-vindas**, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças para garantir **vantagem competitiva** ao cliente.
3. **Entregar frequentemente** software funcional, de poucas semanas a poucos meses, com preferência à menor escala de tempo.
4. **Pessoas de negócio e desenvolvedores** devem trabalhar em conjunto diariamente durante todo o projeto.
5. Construir projetos em torno de **indivíduos motivados**. Dando a eles o ambiente e suporte necessário e confiar que farão o trabalho.
6. A forma mais eficiente e eficaz de comunicar informações para dentro e através de uma equipe de desenvolvimento é através de **conversa face a face**.



# Manifesto Ágil: Princípios 6-12

---

1. **Software funcional** é a principal medida de **progresso**.
2. Processos ágeis promovem um **ambiente sustentável**. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
3. Contínua **atenção à excelência técnica** e bom design aumenta a agilidade.
4. **Simplicidade**, a arte de maximizar a quantidade de trabalho não realizado, é essencial.
5. As melhores arquiteturas, requisitos e designs emergem de **equipes auto-organizáveis**.
6. Em intervalos regulares, **a equipe reflete sobre como se tornar mais eficaz**, então, ajusta e otimiza seu comportamento.





# Manifesto Ágil: Princípios

---

Em resumo, **os 12 princípios** enfatizam principalmente:

- Adaptação a mudanças;
- Entrega iterativa;
- Colaboração eficaz;
- Equipes capacitadas;
- Foco na eficiência;
- Software funcional, sustentabilidade;
- E a entrega contínua de valor ao cliente.



# Manifesto Ágil: Consequências

---

Com o lançamento e a difusão do Manifesto Ágil, surgiram várias evoluções na indústria de desenvolvimento de software:

- **Adoção de Métodos Ágeis:** Frameworks como Scrum, Kanban e XP são baseados nos princípios do Manifesto Ágil, promovendo colaboração, entrega contínua e adaptação a mudanças;
- **Transformação Organizacional:** Empresas adotam métodos ágeis para se adaptar mais rapidamente às mudanças do mercado e responder às necessidades dos clientes;
- **Inovação Contínua:** A abordagem ágil encoraja a experimentação e iteração constante, fomentando a inovação contínua. Isso possibilita a geração de novas ideias, a identificação rápida de soluções mais eficientes e a adaptação a novos requisitos e tendências de mercado.







# Métodos Ágeis

# Métodos Ágeis

---

Como vimos, uma das consequências do Manifesto Ágil foi a **Adoção de Métodos Ágeis** por grande parte das empresas:

- Em suma, **os métodos ágeis consistem em uma abordagem de desenvolvimento de software que prioriza interações colaborativas, flexibilidade e entrega contínua de valor.**
- Por compartilharem dos valores e princípios, os métodos ágeis são comumente diretamente associados ao Manifesto Ágil;



# Métodos Ágeis

---

- Contudo, **o manifesto ágil não cita diretamente as metodologias que devem ser adotadas;**
- E algumas das mais conhecidas, como por exemplo o Kanban, surgiram inclusive antes do manifesto ágil ser elaborado;
- Entre as principais metodologias ágeis estão o Scrum, Kanban, XP (Extreme Programming), Lean, entre outros. Cada um com suas particularidades, **mas todos focados em agilidade, colaboração e melhoria contínua.**



# Métodos Ágeis: Onde aplicar

---

Apesar do nosso foco ser a aplicação de métodos ágeis em um contexto de tecnologia e software, é importante saber que **a aplicação pode ir além desse cenário.**

- Contextos favoráveis para métodos ágeis **envolvem produtos inovadores, experimentação contínua e também ambientes complexos**, onde os requisitos ainda estão abstratos e podem mudar rapidamente;
- O que pode ocorrer em indústrias de tecnologia, produção, setores criativos, etc;
- De maneira geral, **são aplicáveis em diversos setores, especialmente onde a flexibilidade e a resposta rápida são essenciais.**



# Métodos Ágeis: Vantagens

## Benefícios dos Métodos Ágeis

### 1. Adaptabilidade

Foco na entrega de valor para o cliente na construção do projeto. Etapas do processo em ciclos curtos, permite aprovação mais rápida das entregas.

### 2. Flexibilidade e produtividade

Sem processos burocráticos, as equipes de produção têm maior autonomia e procedimentos mais simplificados.

### 3. Colaboração entre equipes

Envolver equipes é o objetivo dos métodos ágeis. Squads e grupos multidisciplinares devem trabalhar em conjunto na busca de soluções em um ambiente colaborativo e motivador.

### 4. Comunicação

Outra vantagem é a comunicação do início ao fim do projeto. É possível estabelecer o diálogo e troca de ideias entre as pessoas e evitar ambiguidades durante o processo.



# Métodos Ágeis:

METODOLOGIAS ÁGEIS			
	PARA QUEM	PRÓS	CONTRAS
SCRUM	Times de <b>qualquer especialidade</b> , a partir de duas pessoas	<b>Versátil</b> , é o sistema <i>agile</i> mais usado no mundo.	Pode ser um <b>complicador para projetos individuais</b> ou muito sucintos
KANBAN	<b>Projetos simples e de fácil mapeamento</b> , inclusive individuais	<b>Simples</b> , baseado no método Get Things Done	Não contempla especificidades, <b>pode ser simples demais</b>
LEAN	Ideias que precisam ser <b>colocadas em prática rapidamente</b>	Prioriza <b>agilidade e economia</b> , muito usado por startups	De <b>difícil execução em companhias tradicionais</b>
SMART	<b>Equipes que buscam mapear e classificar</b> itens ou etapas de processos	Pode ser aplicado a <b>qualquer etapa de qualquer processo</b>	Requer <b>dados e profundidade de análise</b>
XP	<b>Engenheiros e desenvolvedores de software</b>	Tem foco na <b>melhoria de comunicação</b> , com princípios bem definidos	Requer o <b>comprometimento real</b> de todas as partes
FDD	<b>Engenheiros e desenvolvedores de software</b>	<b>Divide tarefas</b> em funcionalidades específicas	<b>Não resolve fluxo de comunicação</b>
MSF	<b>Engenheiros e desenvolvedores de software</b>	<b>Testado e aprovado</b> , tem anos de uso por equipes de dev	<b>Defasado</b> em relação a métodos mais novos



# Conceitos importantes em ambientes ágeis

---

Antes de falarmos especificamente de alguma metodologia, é interessante ver **conceitos gerais importantes que são comumente encontrados em ambientes que aplicam a prática do desenvolvimento ágil**, como por exemplo:

- CI/CD;
- Squad ou Equipe;
- Task ou Tarefa;
- Board ou Quadro de tarefas;
- User stories ou História de usuários;
- Backlog de Produto;



# CI/CD

---

## CI/CD:

- **Integração Contínua (CI):**
  - Processo automatizado de integração frequente de código.
  - Testes automáticos realizados regularmente para identificar problemas rapidamente.
- **Entrega Contínua (CD):**
  - Amplia a CI para automatizar a entrega de código ao ambiente de produção.
  - Entrega de incrementos funcionais rapidamente e com qualidade.
- **Relação com Ambientes Ágeis:**
  - CI/CD permite a implementação eficaz dos princípios ágeis.
  - Facilita **entregas frequentes, adaptabilidade e resposta rápida às mudanças.**
  - Agiliza o ciclo de desenvolvimento.
  - Garante consistência, qualidade e estabilidade.
  - Permite a rápida iteração e feedback constante.

# Squad

---

## Squad:

- Equipe **multidisciplinar e autogerenciada**.
- Responsável por uma parte específica do produto ou funcionalidade.
- Diversidade de habilidades (desenvolvedores, designers, especialistas em QA, etc.).
- Foco em objetivos claros e entregas de valor ao usuário.
- **Relação com Ambientes Ágeis:**
  - Rapidez na tomada de decisões.
  - Maior flexibilidade e adaptabilidade às mudanças.
  - Melhoria na comunicação e colaboração entre os membros da equipe.

# Task

---

## Task:

- Tarefas específicas e menores dentro de uma história de usuário.
- Geralmente são unidades de trabalho realizáveis em um curto período.
- Dividem o trabalho em partes menores e mais gerenciáveis.
- **Relação com Ambientes Ágeis:**
  - Contribuem para a **entrega incremental** de funcionalidades.
  - Permitem um gerenciamento mais preciso do progresso.

# Board

---

## Board:

- Ferramenta visual para **gerenciar o fluxo de trabalho**.
- Divisão das tarefas em colunas que representam os estágios do processo.
- Exemplos: Jira Board, Trello, KanbanFlow;
- **Relação com Ambientes Ágeis:**
  - É uma ferramenta essencial em metodologias ágeis como o Kanban e Scrum.
  - Proporciona transparência e facilita a colaboração da equipe.
  - Visualização clara do fluxo de trabalho e das pendências.

# User Stories

---

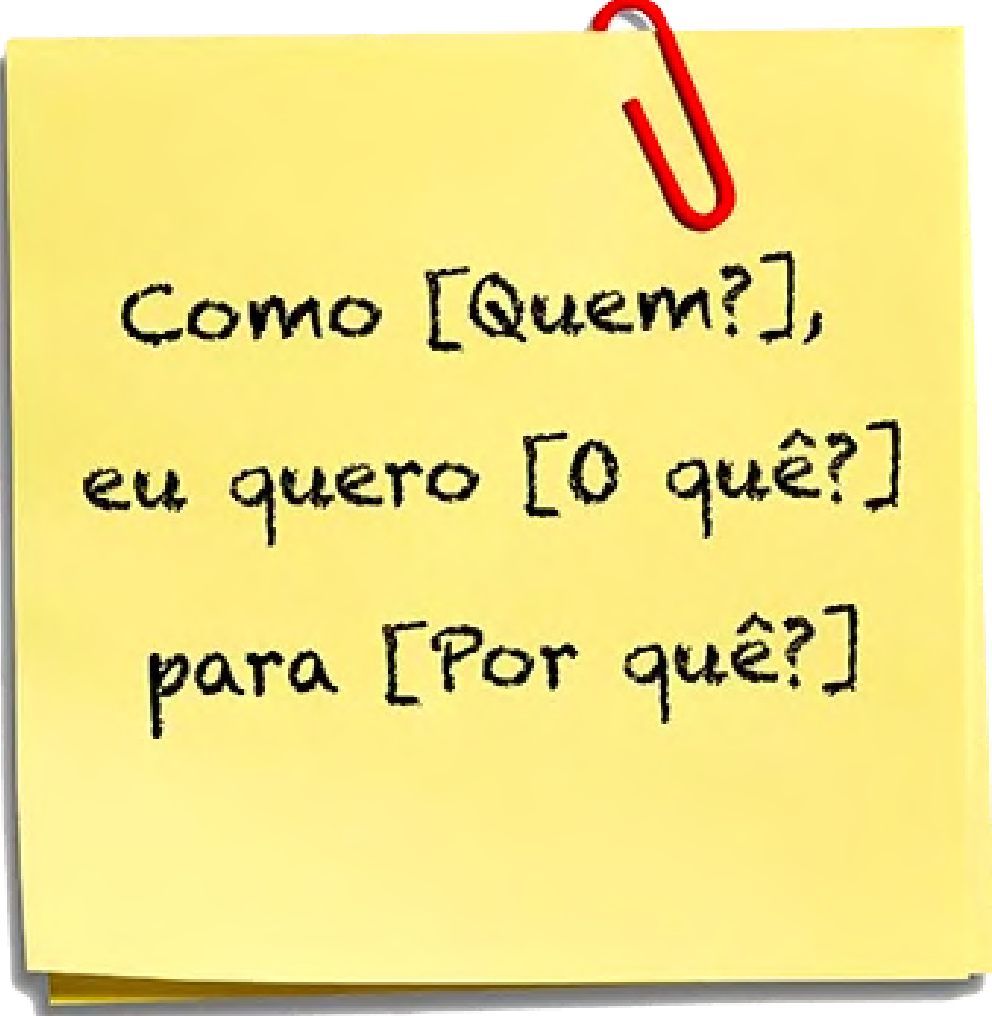
## User Stories:

- Breves descrições de uma **funcionalidade do ponto de vista do usuário**.
- Expressam o que o usuário precisa e por quê, não detalhando o "como".
- **Relação com Ambientes Ágeis:**
  - User stories são a base para definir requisitos em metodologias ágeis como o Scrum, mas não são de uso exclusivo de nenhuma metodologia.
  - Permitem uma comunicação clara e uma compreensão compartilhada do que deve ser entregue.
  - Foco na entrega de valor ao usuário.
  - Flexibilidade para adaptar-se a mudanças nas necessidades do cliente.

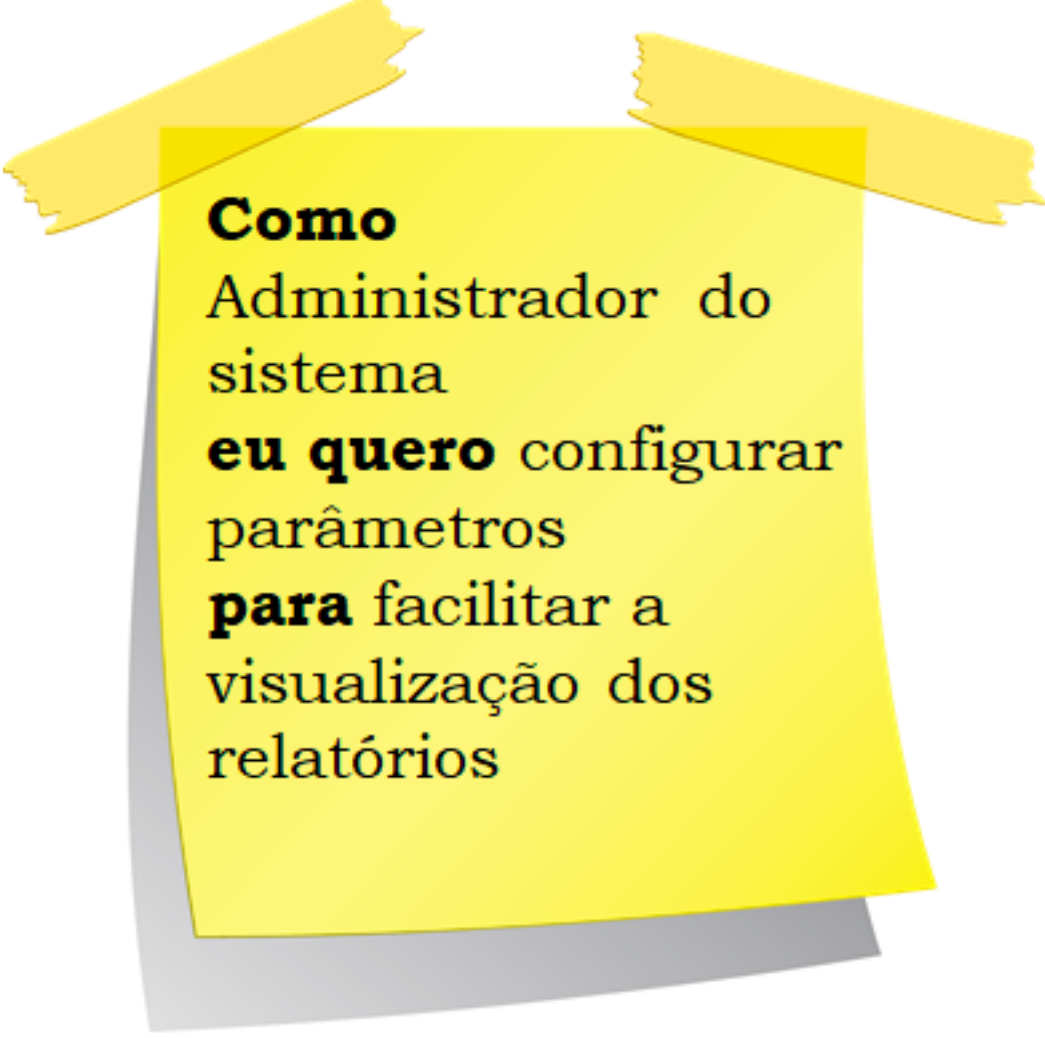


# User Stories

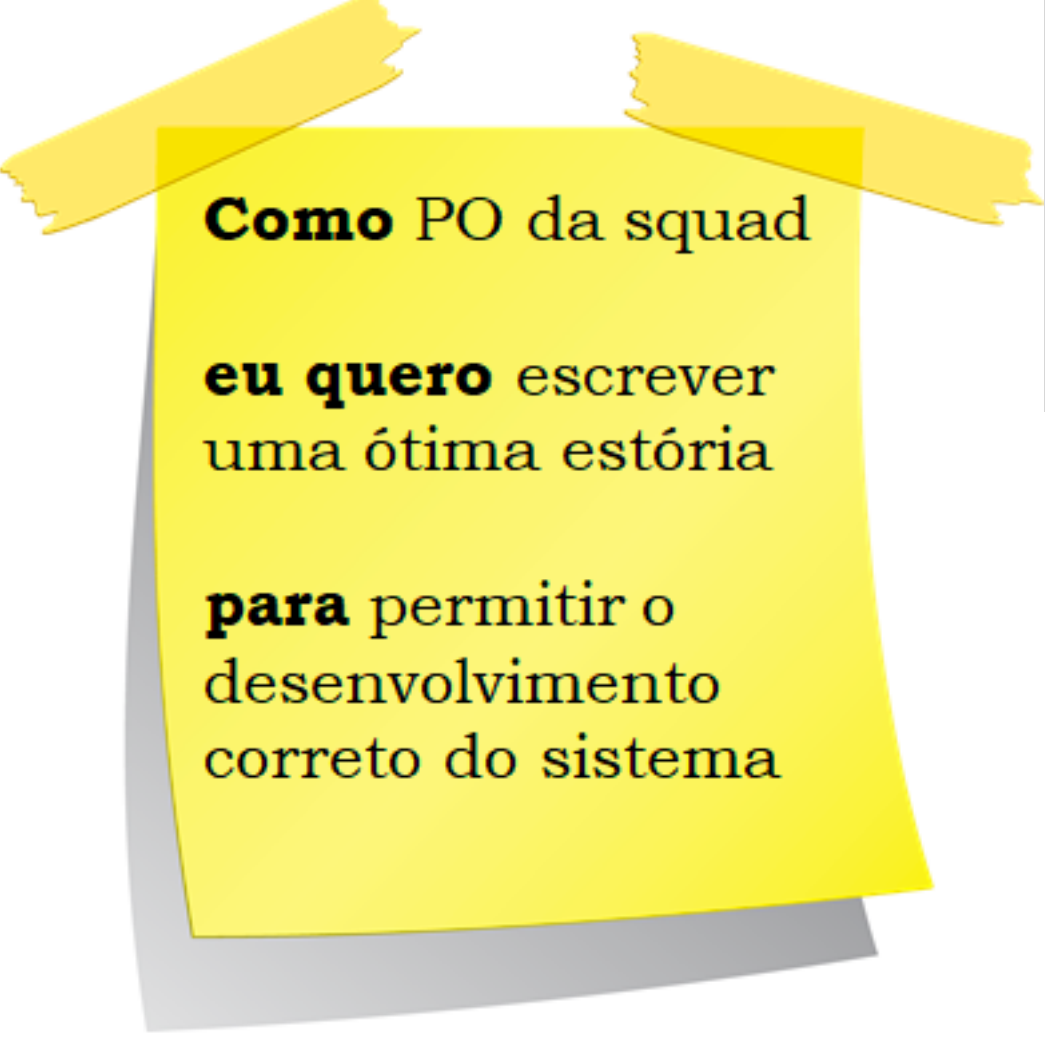
## Exemplo de User Stories:



Como [Quem?],  
eu quero [O quê?]  
para [Por quê?]



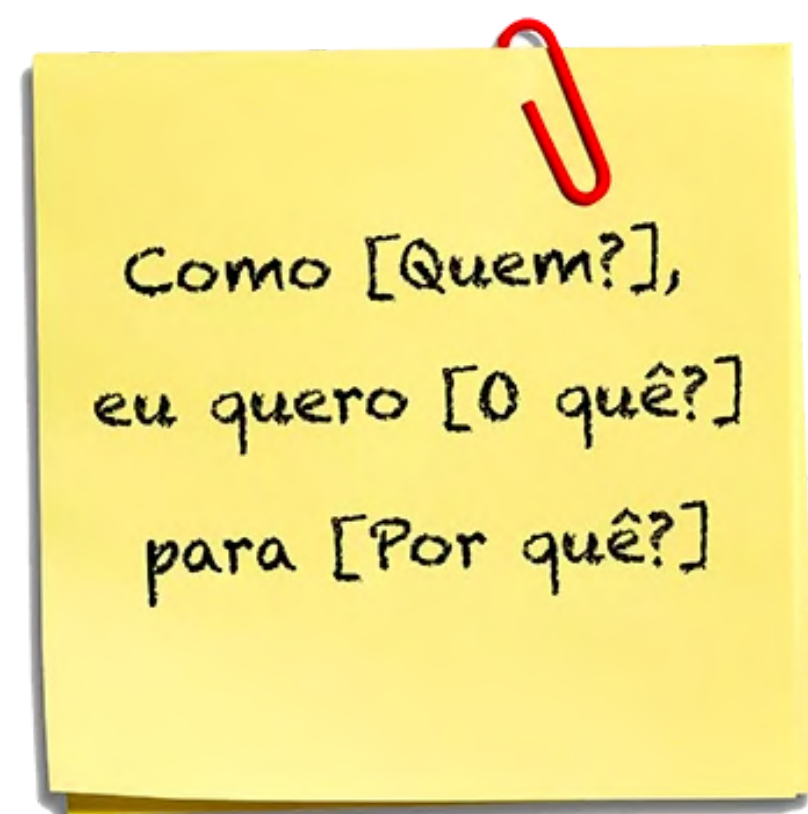
**Como**  
Administrador do  
sistema  
**eu quero** configurar  
parâmetros  
**para** facilitar a  
visualização dos  
relatórios



**Como** PO da squad  
**eu quero** escrever  
uma ótima estória  
**para** permitir o  
desenvolvimento  
correto do sistema

# User Stories

## Exemplo de User Stories:



Como um <ator>,  
eu quero / preciso / gostaria de <ação>  
para que <objetivo>

### Cartão de história de usuário

**Nome:** Consulta de Livro.  
**Como** um vendedor,  
**eu quero** procurar por livros filtrando  
por nomes  
**para que** seja possível  
verificar se existe o livro em estoque  
para venda.

**Ator:** é o proprietário da história, quem  
irá utilizar a funcionalidade ou requisito.  
(perfil do usuário).

**Ação:** é o que o ator quer fazer dentro  
do sistema, esperando que o objetivo  
seja alcançado.

**Objetivo** é o que o ator espera que vá  
acontecer após a ação ser executada.  
Também vista como uma justificativa.

# User Stories

## Principais características das User Stories:

- **Centradas no Usuário:**
  - As user stories se concentram nas necessidades e no ponto de vista do usuário. Elas são descritas do ponto de vista do usuário final.
- **Funcionalidade e Valor:**
  - Descrevem uma funcionalidade específica que **agrega valor** ao usuário ou ao cliente.
- **Pequenas e Gerenciáveis:**
  - São pequenas o suficiente para serem implementadas dentro de um curto espaço de tempo, geralmente em um ou poucos ciclos de desenvolvimento.
- **Não Detalham o Como:**
  - Focam no "o que" e "por quê", não no "como". Não detalham a implementação técnica, deixando espaço para a equipe decidir como implementar a funcionalidade.
- **Independentes e Estimáveis:**
  - Devem ser independentes entre si e estimáveis, ou seja, a equipe deve ser capaz de estimar o tempo necessário para implementar cada user story, mesmo que seja necessário quebrá-la em tarefas menores.

*Follow the INVEST guidelines for good user stories!*



I	independent
N	negotiable
V	valuable
E	estimable
S	small
T	testable

# Backlog de Produto

---

## Backlog de Produto:

- Lista dinâmica de funcionalidades, melhorias e requisitos necessários para o produto.
- Priorizada e gerenciada pelo **Product Owner**, ou por outra persona que desempenha o papel de PO no time.
- Evolui constantemente com base no feedback do cliente e nas mudanças do mercado.
- Contém itens como user stories, tarefas ou requisitos.
- **Relação com Ambientes Ágeis:**
  - É um elemento fundamental em metodologias ágeis como o Scrum, Kanban, etc.
  - Facilita a priorização e a entrega incremental de valor ao cliente.
  - Permite adaptabilidade às mudanças e priorização contínua.
- Importante não confundir com **Backlog de Sprint**, que veremos mais na frente nessa disciplina, e pertence a metodologias mais específicas.

# User Stories no board Jira: Prática 01

## Vamos praticar?



1. **Todos:** acessar o site da Atlassian (dona do Jira Board) e criar uma conta **free**;
2. **Um (1) membro** de cada equipe deve criar um novo projeto com o template de **SCRUM**, e convidar os demais colegas de equipe;
  - a. Quanto ao nome do projeto, deve-se nomear com o padrão: WA-{numeroGrupo}
3. Baseado na descrição do sistema disponível no *colabweb*, deve-se:
  - a. **Elaborar no mínimo seis (6) user stories**, levando-se em consideração as principais *features* descritas, e as boas práticas de construção de user stories vistas em aula;
  - b. Para cada user story, deve ser criado um novo item do tipo "Story" no backlog do board do Jira, criado no passo anterior (2).
    - i. No título, deve haver um resumo do conteúdo da User Story;
    - ii. Na descrição, deve haver a User Story em si, no formato Quem / O que? / Por quê?



# Métodos Ágeis: SCRUM

Parte 1



# Scrum

## Sobre Scrum

### O que é Scrum?

Scrum ajuda pessoas e equipes **a entregar valor** de forma incremental e **colaborativa**. Se você está apenas começando, pense nisso como uma forma de realizar o trabalho em equipe, em pequenas partes de cada vez, com experimentação e ciclos de feedback ao longo do caminho.

Saber mais



# Scrum

---

- **Scrum** é uma **estrutura ágil de gerenciamento de projetos**;
- Com foco na **flexibilidade e adaptação contínua**;
- Que tem como **princípios**: Transparência, Inspeção, Adaptação;
- E geralmente é praticado por um **time multidisciplinar**.
- Com os **objetivos** de:
  - Entregar valor constantemente.
  - Adaptar-se às mudanças do mercado.
  - Melhoria contínua do produto e processo.



# Scrum: Princípios

## The 6 key Scrum principles



Control over  
empirical processes



Self-organization



Collaboration



Value-based  
prioritization



Time-boxing



Iterative development

# Scrum: Princípios

---

- **Decisões com Base Empírica:**
  - Decisões fundamentadas em dados e experiências reais, não em suposições.
  - Envolve três ideias: Transparência, Inspeção e Adaptação.
- **Autogerenciamento:**
  - Time responsável por definir como alcançar objetivos e resolver desafios.
- **Colaboração:**
  - Trabalho conjunto e comunicação contínua para atingir metas.
- **Prioriza a Entrega de Valor:**
  - Foco constante em entregar o máximo valor possível ao cliente.
- **Time-Boxing:**
  - Definição de prazos limitados (sprints) para as atividades, promovendo foco e entrega frequente.
- **Processo Iterativo:**
  - Abordagem de desenvolvimento em ciclos repetitivos para melhorias contínuas.



# Obrigada!

---

Dúvidas?

- **Slack**
- **Email:** [jlslc@icomp.ufam.edu.br](mailto:jlslc@icomp.ufam.edu.br)

