



# Desenvolvimento de **Sistemas**

**Aula 02 – Metodologias de  
Desenvolvimento e Framework Scrum**

Lorrany B A Marim

**SENAI**



## Por que falar de métodos de desenvolvimento?

Na prática, muitas equipes começam um projeto cheias de energia, fazem um plano gigantesco... e, meses depois, percebem que:

- O cliente mudou de ideia.
- A tecnologia mudou.
- Os requisitos eram incompletos ou mal entendidos.
- Consertar o que foi feito custa caro e leva tempo.



## Efeito Túnel

No modelo tradicional, muitas vezes:

- Passamos **muito tempo só planejando e documentando**.
- O cliente **só vê o sistema funcionando no final**.
- Fica todo mundo “no escuro”, sem ter certeza se o caminho está certo.

É como entrar em um **túnel longo**: você entra lá com um plano, anda, anda, anda... e só vê a luz (o resultado) depois de meses.

Se ao sair do túnel o cliente diz: “Não era bem isso que eu queria...”, o prejuízo é grande.



# A Incerteza no Desenvolvimento

Diferente de construir uma casa, no software:

- Não sabemos **todos os detalhes** desde o início.
- O **negócio muda** (nova lei, concorrente, prioridade).
- O **cliente aprende durante o projeto** e muda de ideia.
- Novas tecnologias surgem no meio do caminho.





# A Incerteza no Desenvolvimento

Tentar prever tudo com precisão absoluta gera:

- **Desperdício** (muito tempo gasto com coisas que serão jogadas fora).
- **Frustração** (equipe e cliente insatisfeitos).
- **Sistema que não atende mais a realidade.**

Daí vem a pergunta:

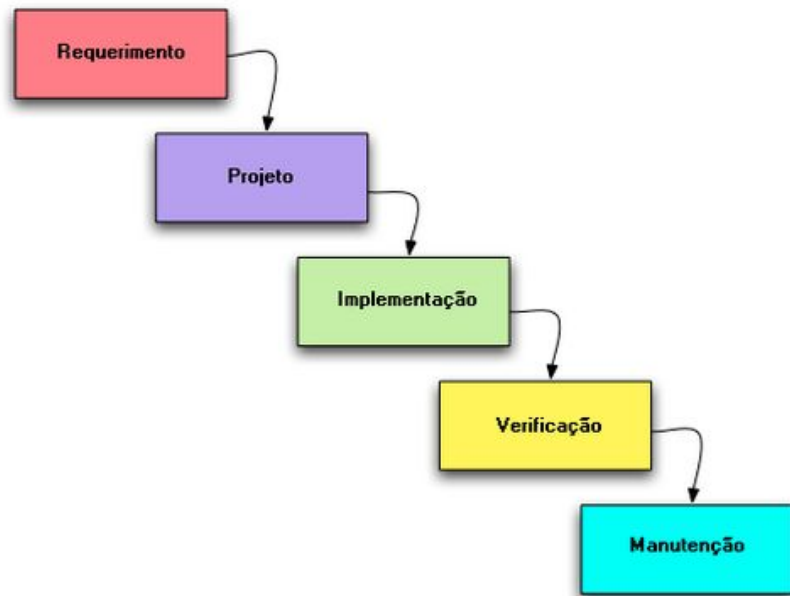
“Como desenvolver software aceitando que mudanças vão acontecer?”



## O Modelo Cascata

O **Modelo em Cascata** é um dos primeiros modelos de ciclo de vida de software. Ele é:

- **Sequencial e rígido** – segue uma ordem fixa, como linha de montagem.
- Bem parecido com projetos de **engenharia civil** (prédio, ponte etc.).





# Fases do Modelo em Cascata

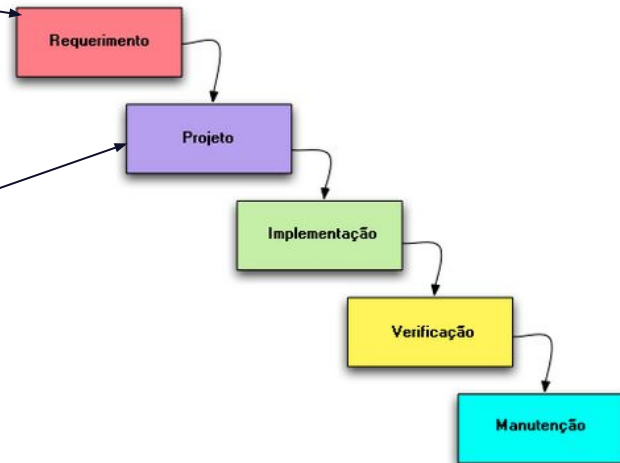
De forma simplificada, as fases são:

## 1. Levantamento de Requisitos

- Entender o que o sistema deve fazer.
- Conversar com o cliente, listar funcionalidades, regras, relatórios etc.

## 2. Análise e Projeto (Design)

- Decidir **como** o sistema vai funcionar internamente.
- Desenhar telas, modelos de dados, arquitetura, diagramas.





# Fases do Modelo em Cascata

## 3. Implementação (Codificação)

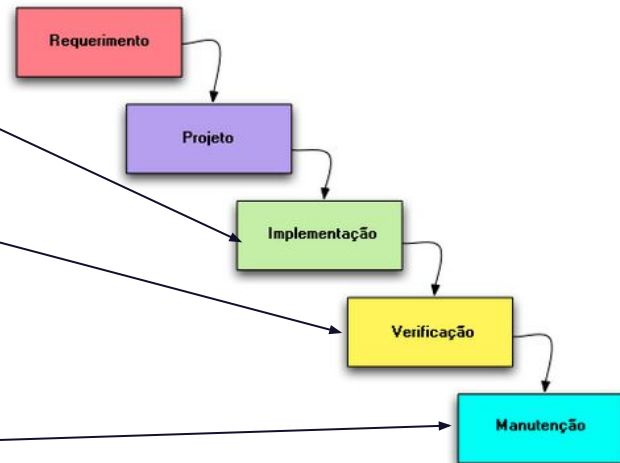
- Programar o sistema de acordo com o que foi definido no projeto.

## 4. Testes

- Verificar se o sistema funciona corretamente e atende aos requisitos.

## 5. Implantação (Entrega)

- Colocar o sistema em produção para o cliente usar.







## Problemas do Modelo em Cascata

- **Planejamento pesado:** tenta-se detalhar e documentar tudo no início, antes de qualquer código.
- **Efeito túnel:** o cliente só vê o sistema pronto lá na frente – feedback chega tarde.
- **Pouca flexibilidade:**
  - Mudanças de escopo são difíceis e caras.
  - Voltar para ajustar fases iniciais (requisitos, design) é complexo.
- **Risco alto:**
  - Se errarmos no começo, o erro só aparece no final, quando já foi gasto muito tempo e dinheiro.

Conclusão: o modelo em Cascata funciona melhor em cenários **muito estáveis**, com poucas mudanças. Em software moderno, isso é raro.



# Manifesto Ágil

Diante dos problemas dos processos pesados e burocráticos, em **2001** um grupo de desenvolvedores se reuniu e criou o **Manifesto Ágil**, propondo uma maneira diferente de enxergar projetos de software.

## Origem:

- Profissionais experientes estavam insatisfeitos com:
  - Muita documentação que ninguém lia.
  - Pouco foco em software de fato funcionando.
  - Pouca colaboração real com o cliente.
  - Dificuldade de lidar com mudanças.





# Os 4 Valores do Manifesto Ágil

Eles se uniram e definiram **4 valores principais**.

**Mais valor à esquerda do que à direita:**

1. **Indivíduos e interações**

**mais que** processos e ferramentas

→ Pessoas conversando e colaborando resolvem mais problemas do que apenas seguir um processo rígido.

2. **Software em funcionamento**

**mais que** documentação abrangente

→ Melhor ter um sistema simples funcionando do que 200 páginas de documentação sem nada pronto.



# Os 4 Valores do Manifesto Ágil

## 3. Colaboração com o cliente

**mais que** negociação de contratos

→ Trabalhar junto com o cliente, ajustando o que for necessário, vale mais do que ficar preso a um contrato inflexível.

## 4. Responder a mudanças

**mais que** seguir um plano

→ É mais importante adaptar o projeto à realidade do que seguir cegamente um plano que ficou desatualizado.

- Importante: os itens da direita **não são descartáveis** (documentação, contratos, plano).
- O que o Manifesto diz é que, **em caso de conflito**, damos mais valor aos itens da esquerda.



# Introdução ao Scrum

O Scrum é uma das principais formas de aplicar os princípios ágeis na prática.

## O que é Scrum?

- Um **framework leve** para gerenciar o desenvolvimento de produtos complexos.
- Ele não diz **como programar**, mas **como organizar o trabalho do time**.
- Ajuda a:
  - Dividir o trabalho em partes menores.
  - Entregar valor de forma contínua.
  - Adaptar o projeto sempre que necessário.



# Origem do Scrum





# Iterativo e Incremental

O Scrum trabalha com:

- **Ciclos curtos** de trabalho, chamados **Sprints** (geralmente de 2 a 4 semanas).
- A cada Sprint, é entregue um **incremento** do produto:
  - Algo que já funciona.
  - Já pode ser testado e avaliado pelo cliente.

## SCRUM SPRINT INFOGRAPH





## Analogia do Bolo

- No modelo tradicional:
  - Fazemos **toda a massa do bolo**, depois **todo o recheio**, depois **toda a cobertura**.  
O cliente só prova no final – se não gostar, é tarde.
- No Scrum:
  - Entregamos o bolo **fatia por fatia**.
  - Cada fatia já vem com massa + recheio + cobertura.  
O cliente já consegue provar e dar feedback:
    - “O recheio está muito doce”
    - “Quero mais cobertura de chocolate”
  - Ajustamos as próximas fatias com base nesse retorno.





# Empirismo e Foco em Valor

O Scrum é baseado em:

- **Empirismo:** aprender com a experiência prática.
  - Fazemos, olhamos o resultado, medimos, ajustamos.
- **Foco em valor:**
  - Não fazemos “qualquer coisa”.
  - Priorizamos aquilo que traz **mais retorno** e **mais valor** para o cliente o mais rápido possível.



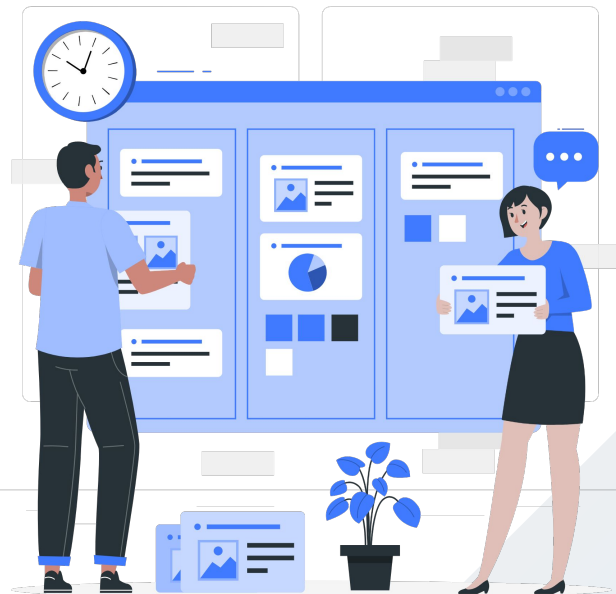


# Pilares do Scrum

Para o Scrum funcionar bem, precisamos de três pilares principais:

## 1. **Transparência**

- Todos sabem o que está sendo feito e em que estágio está.
- As informações importantes estão visíveis:
  - Quadro de tarefas (Kanban / Scrum Board).
  - Quem está fazendo o quê.
  - O que está pronto e o que está bloqueado.





# Pilares do Scrum

## 2. Inspeção

- O time verifica frequentemente:
  - O produto (o que foi desenvolvido).
  - O processo (como estamos trabalhando).
- Isso acontece em momentos como:
  - Daily Scrum.
  - Sprint Review.
  - Sprint Retrospective.





# Pilares do Scrum

## 3. Adaptação

- Se percebemos que algo não está bom:
  - Ajustamos rapidamente o plano.
  - Mudamos prioridades no Backlog.
  - Melhoramos a forma de trabalho.
- Não continuamos fazendo algo errado só porque “estava no plano”.





# Pilares do Scrum

## 4. (Complemento importante) Confiança

- O ambiente deve permitir que o time:
  - Aponte problemas.
  - Sugira melhorias.
  - Admita erros sem medo.
- Sem confiança, a transparência e a adaptação não acontecem de verdade.





# Papéis no Scrum

No Scrum não existe “chefe mandando tarefas” no dia a dia.

Existem três papéis principais:

## Product Owner (PO):

- É a **voz do cliente**.
- Focado em “**o que**” deve ser feito.

Responsabilidades:

- Definir e **priorizar** as funcionalidades que trazem mais valor.
  - Gerenciar o **Product Backlog** (lista de tudo o que o produto pode ter).
  - Estar disponível para tirar dúvidas do time.
- Ele se preocupa com:
  - Retorno sobre investimento.
  - Entregar primeiro o que é **mais importante**.



# Papéis no Scrum

## Scrum Master (SM)

- É um **líder servidor** e **facilitador**.
- Não é “chefe” nem “gerente de projeto tradicional”.

Responsabilidades:

- Ajudar o time a entender e seguir o Scrum.
- Remover **impedimentos** (coisas que atrapalham o trabalho).
- Proteger o time de interrupções desnecessárias.
- Facilitar as cerimônias (reuniões) do Scrum.



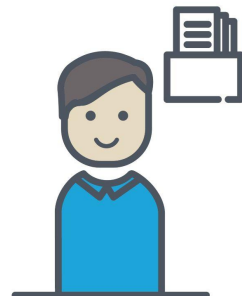


# Papéis no Scrum

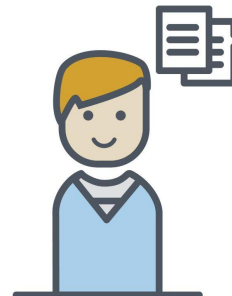
## Time de Desenvolvimento (Dev Team)

- Grupo de profissionais que **constroem o produto**.
- É um time **multidisciplinar**:
  - Pode ter devs back-end, front-end, testadores, UX, etc.
- Responsabilidades:
  - Definir **"como"** o trabalho será feito tecnicamente.
  - Estimar esforço.
  - Se auto-organizar para entregar o que foi combinado na Sprint.
- Características:
  - **Auto-organizado**: o próprio time decide como dividir as tarefas.
  - **Sem um "chefe" interno**: todos são responsáveis pelo resultado.





PRODUCT OWNER

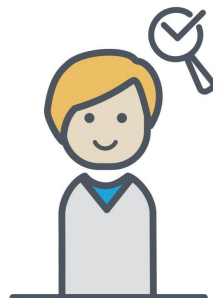


SCRUM MASTER

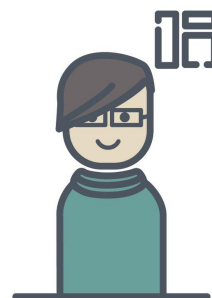
# SCRUM TEAM



DEVELOPER



QA



UX/UI DESIGNER



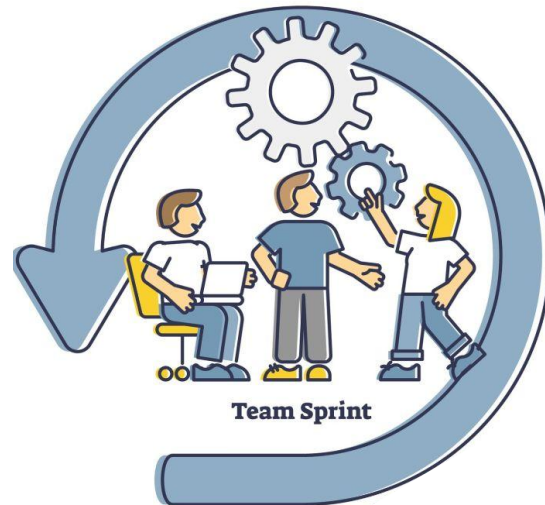
# Eventos Scrum (Cerimônias)

Os eventos do Scrum dão **ritmo** ao trabalho.

## Sprint

- É o **coração do Scrum**.
- Período de tempo **fixo** (timebox), geralmente de 2 a 4 semanas.  
Durante a Sprint:
  - O time trabalha em um conjunto de itens do Backlog.
  - Ao final, deve haver um **incremento do produto** "Pronto".

## SCRUM





# Artefatos do Scrum

Os artefatos são os “documentos vivos” que dão visibilidade ao trabalho.

## Product Backlog

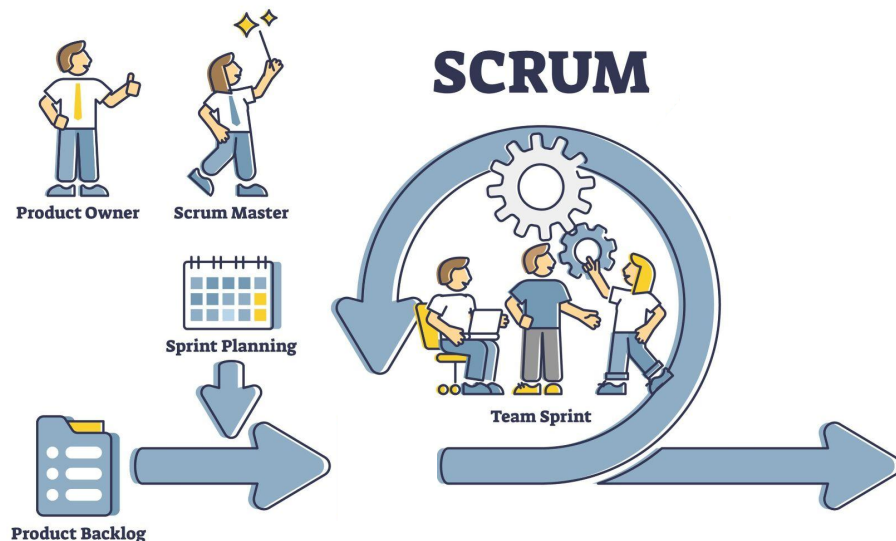
- Lista **ordenada** de tudo que o produto pode ter:
  - Funcionalidades.
  - Correções.
  - Melhorias técnicas.
- Características:
  - Está sempre **em evolução** (nunca “fechado”).
  - Cada item tem:
    - Descrição.
    - Prioridade.
    - Uma estimativa (quando possível).
- Responsável: **Product Owner**.



# Eventos Scrum (Cerimônias)

## Sprint Planning

- Reunião de **planejamento da Sprint**.
- Perguntas principais:
  1. **O que** vamos entregar nesta Sprint?
  2. **Como** vamos fazer esse trabalho?
- Participam: Product Owner, Scrum Master e Time de Desenvolvimento.

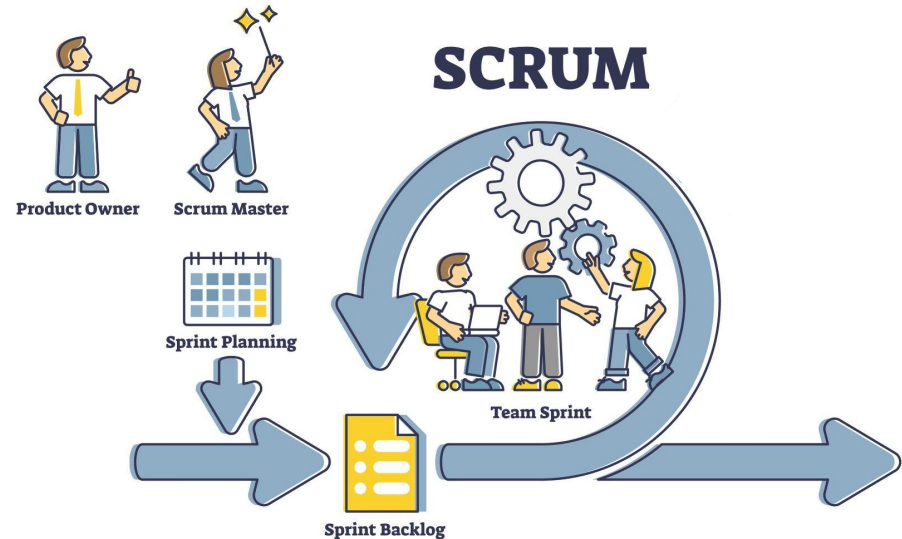




# Artefatos do Scrum

## Sprint Backlog

- Conjunto de itens do Product Backlog escolhidos para a **Sprint atual**.
- Inclui:
  - Os itens selecionados.
  - O plano do time para entregar esses itens (tarefas, subtarefas).
- Responsável: **Time de Desenvolvimento**.

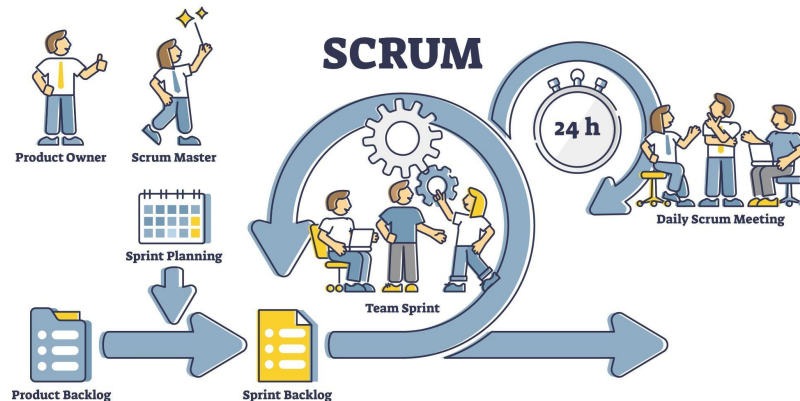




# Eventos Scrum (Cerimônias)

## Daily Scrum (Daily Meeting)

- Reunião **diária e rápida** (até 15 minutos).
- Objetivo:
  - Alinhar o trabalho do dia.
  - Identificar impedimentos.
- Não é reunião de “puxão de orelha” nem “relatório para o chefe”.
- Foco: **colaboração e transparência** dentro do time.

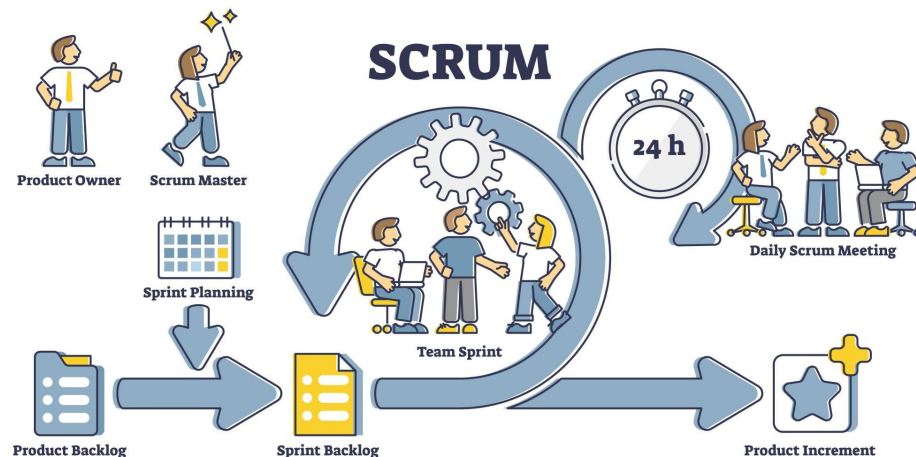




# Artefatos do Scrum

## Incremento

- Resultado tangível da Sprint:
  - A **soma de todos os itens** completados na Sprint atual
    - o valor dos incrementos das Sprints anteriores.
- Deve estar de acordo com a **Definição de Pronto (DoD)**.
- O cliente deve ser capaz de **ver, testar e usar** esse incremento.





# Eventos Scrum (Cerimônias)

## Sprint Review

- Acontece ao **final da Sprint**.
- Objetivo:
  - Mostrar o produto funcionando para o cliente e demais interessados.
  - Coletar **feedback real** sobre o que foi entregue.
- A partir dessa reunião:
  - O Product Owner pode ajustar prioridades do Backlog.
  - Novas ideias podem surgir.







# Eventos Scrum (Cerimônias)

## Sprint Retrospective

- Reunião **interna do time**, também no final da Sprint.
- Objetivo:
  - Discutir **como** foi o trabalho (processo, comunicação, ferramentas).
  - Identificar o que funcionou bem e o que precisa melhorar.
  - Definir pequenas ações de melhoria para a próxima Sprint.





# Artefatos do Scrum

## Definição de Pronto (DoD – Definition of Done)

- Acordo do time sobre o que significa algo estar **“Pronto”**.
- Exemplos (variam de time para time):
  - Código desenvolvido.
  - Testes unitários executados.
  - Testes de integração realizados.
  - Sem bugs conhecidos críticos.
  - Documentação mínima atualizada.
- Importância:
  - Garante um **padrão de qualidade**.
  - Evita que itens sejam considerados “prontos” pela metade.



Obrigado!  
***SENAI***

