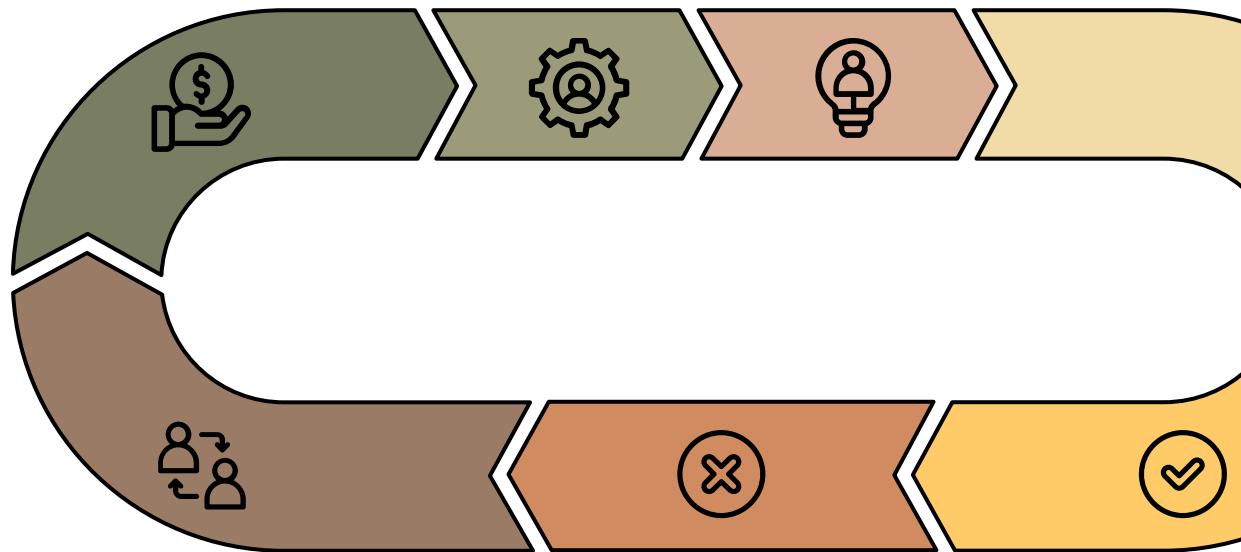


Projeto de Bloco: Desenvolvimento Front-end com Frameworks



Instituto Infnet

Prof. Kennedy Carvalho

Olá {{mundo}} !

Kennedy Carvalho

A.k.a

Engenheiro de Software

Who?

@kndrio

<https://knd.rio>



Sobre a Disciplina

Objetivo da Disciplina

Desenvolver um projeto que une as competências apresentadas nas disciplinas regulares

Tópicos da Aula

Definições

Ciclo de Vida

Metodologias

Boas Práticas

Scrum

Definições

IEEE - Engenharia de Software

Aplicação de um processo sistemático, disciplinado e quantificado ao desenvolvimento, a operação e a manutenção de software.

Pressman - Processo de software

Metodologia para as atividades, ações e tarefas necessárias para desenvolver um software de alta qualidade.

Ciclo de Vida do Software

Sommerville

Especificação

A funcionalidade do software e as restrições a seu funcionamento devem ser definidas.

Projeto e Implementação

O software deve ser produzido para atender às especificações

Validação

O software deve ser validado para garantir que atenda às demandas do cliente

Evolução

O software deve evoluir para atender às necessidades de mudança dos clientes.

Motivos para desenvolver um software



Redução de custos.



Aumento do nível de serviço e desempenho dos recursos humanos.



Melhoria do processo de tomada de decisões pela administração da empresa.



Aumento da vantagem competitiva sobre os concorrentes.



Modelo de negócio onde o software é o principal produto da empresa.



Aumento da qualidade dos produtos e serviços, por meio da automação das rotinas.



Importância da Metodologia

- Falta de qualidade do produto final;
- Falhas na comunicação
- Não cumprimento dos prazos;
- Não cumprimento dos custos.
- Gerenciamento de complexidade;
- Comunicação entre pessoas envolvidas;
- Redução dos custos no desenvolvimento;
- Previsão do comportamento futuro dos sistemas.

Software

Código
RapidMiner
controle

LICENCIAMENTO

IBM
Cerner
Palantir
NATUREZA
Spotify
Proprietário
Educacional

Salesforce

Avast
Firmware
Office
Ciência
Dados
ANSYS
RStudio
Aberto
AutoCAD
Blackboard
S/4HANA
Segurança
Militar
Móvel
System
MacOS
IoT
bibliotecas
treinamento
Embarcado
Weka
McKesson
Open
Whatsapp
Desktop
Source
Visual
STUDIO
Learning
MATLAB
Médico

APLICAÇÃO

Firmware
Office
Ciência
Dados
ANSYS
RStudio
Aberto
AutoCAD
Blackboard
S/4HANA
Segurança
Militar
Móvel
System
MacOS
IoT
bibliotecas
treinamento
Embarcado
Weka
McKesson
Open
Whatsapp
Desktop
Source
Visual
STUDIO
Learning
MATLAB
Médico

PROPOSITO

scikit-learn
Microsoft
Watson
Coursera
PLATAFORMA
Sistemas
Sistema
dispositivos

Adobe

Windows
Python
Web
Comando
Epic
Artificial
Comercial

dispositivos

Netflix
Artificial
Comercial

com

Photoshop
Wordpress
Firefox
Instagram
Linux
Machine
Gotham
McAfee

DOMÍNIO

Científico
Instagran
Linux
Machine
Gotham
McAfee

WorkSpace

Maps
Fortnite
Ubuntu
Segurança
automóveis
Negócios
QuickBooks

Google

Cloud

Control

TensorFlow
Keras
SAP
Simuladores
Kaspersky

Primeiras Metodologias

Os primeiros processos de desenvolvimento de software foram propostos ainda nas décadas de 60 e 70 — eram estritamente sequenciais.

Essa primeira visão de processo era natural, visto que projetos de Engenharia tradicional também são sequenciais e precedidos de um planejamento detalhado.

Começou-se a perceber que software é diferente de outros produtos de Engenharia.

Essa percepção foi ficando clara devido aos problemas frequentes enfrentados por projetos de software nas décadas de 70 a 90.

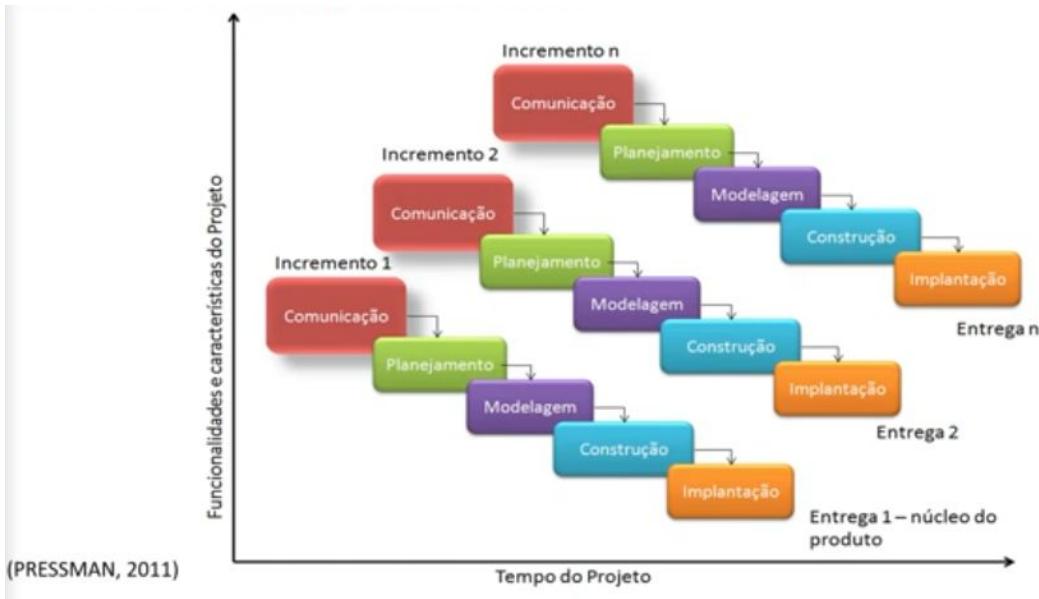


Modelo em Cascata



(SOMMERVILLE, 2011)

Modelo Incremental



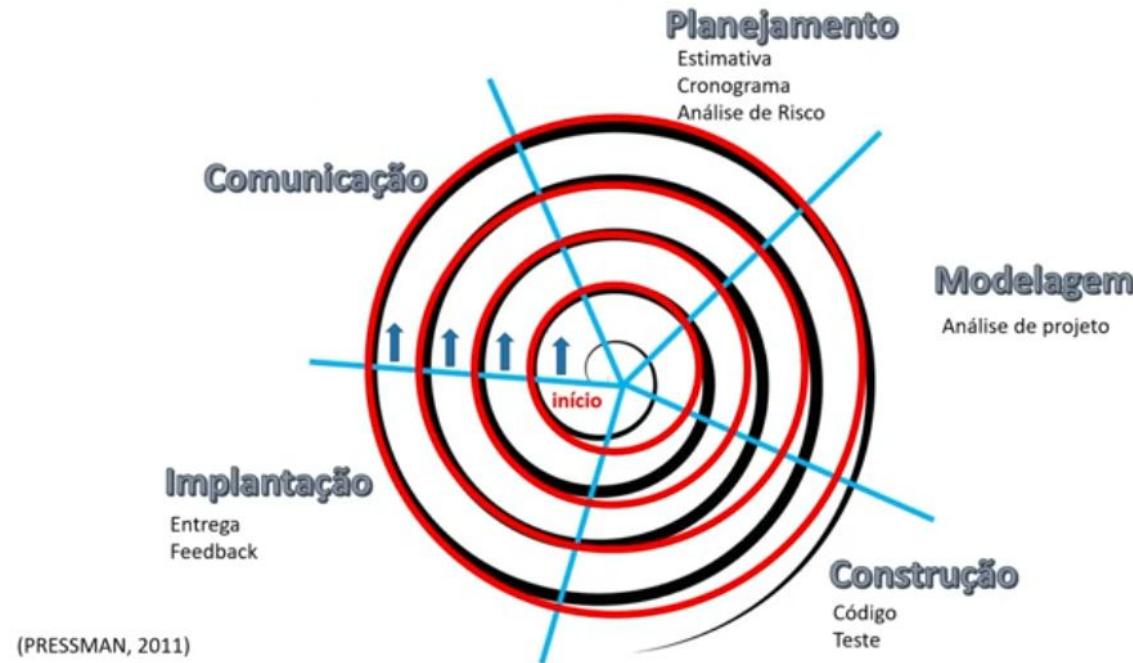
Modelo de Prototipação



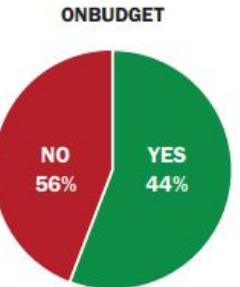
(PRESSMAN, 2011)



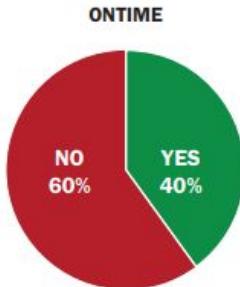
Modelo Espiral ou Evolutivo



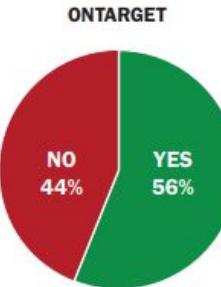
CHAOS REPORT 2015



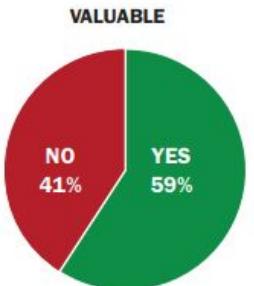
The percentage of projects that were OnBudget from FY2011–2015 within the new CHAOS database.



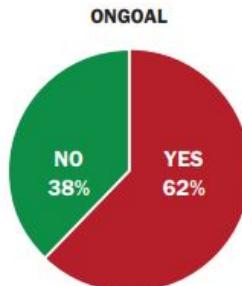
The percentage of projects that were OnTime from FY2011–2015 within the new CHAOS database.



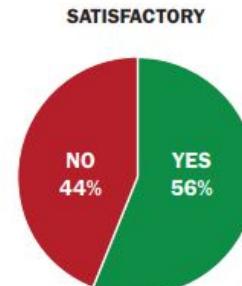
The percentage of projects that were OnTarget from FY2011–2015 within the new CHAOS database.



The percentage of projects considered valuable from FY2011–2015 within the new CHAOS database.

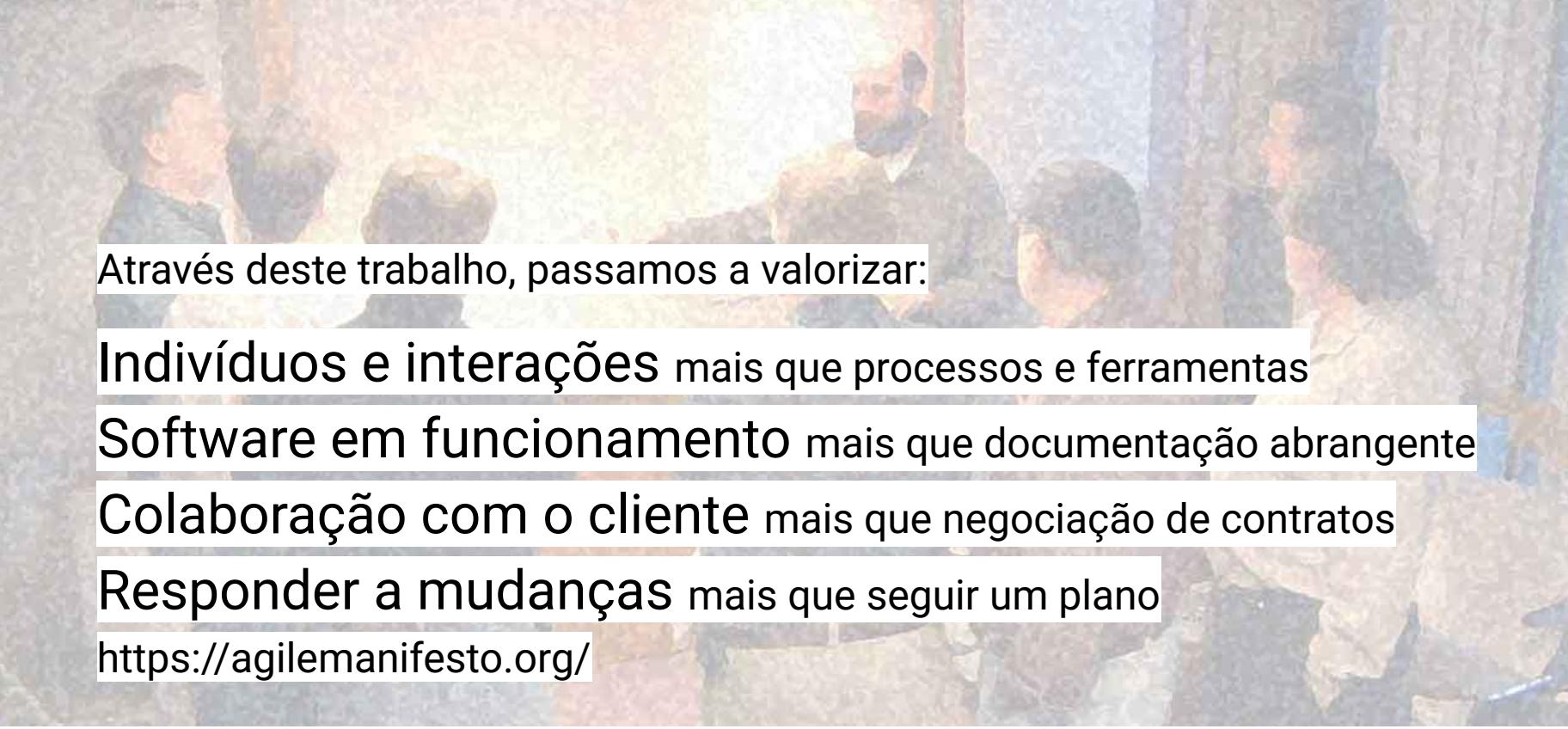


The percentage of projects that were OnGoal from FY2011–2015 within the new CHAOS database.



The percentage of projects considered satisfactory from FY2011–2015 within the new CHAOS database.

Metodologias Ágeis



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

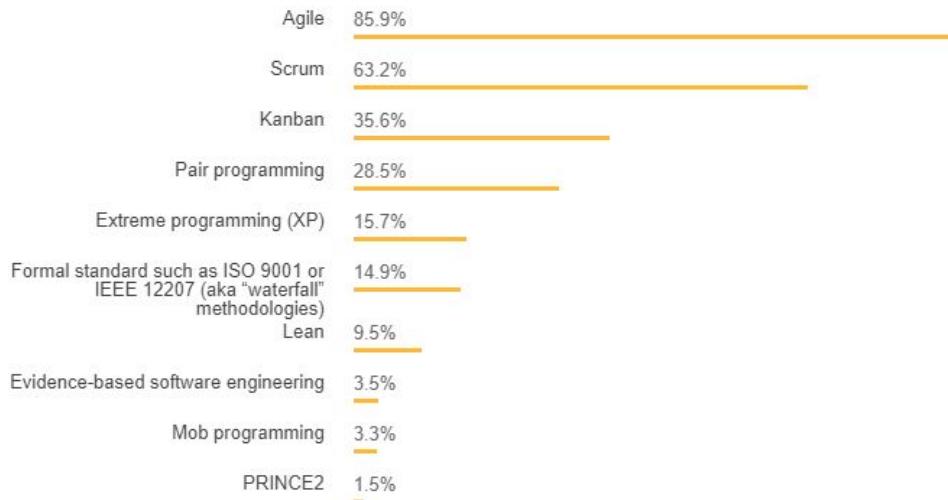
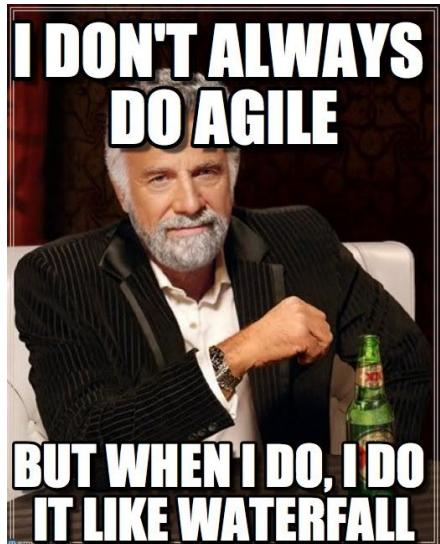
<https://agilemanifesto.org/>

Metodologias Ágeis

Which Methodologies Do Developers Use?

All Respondents

Professional Developers



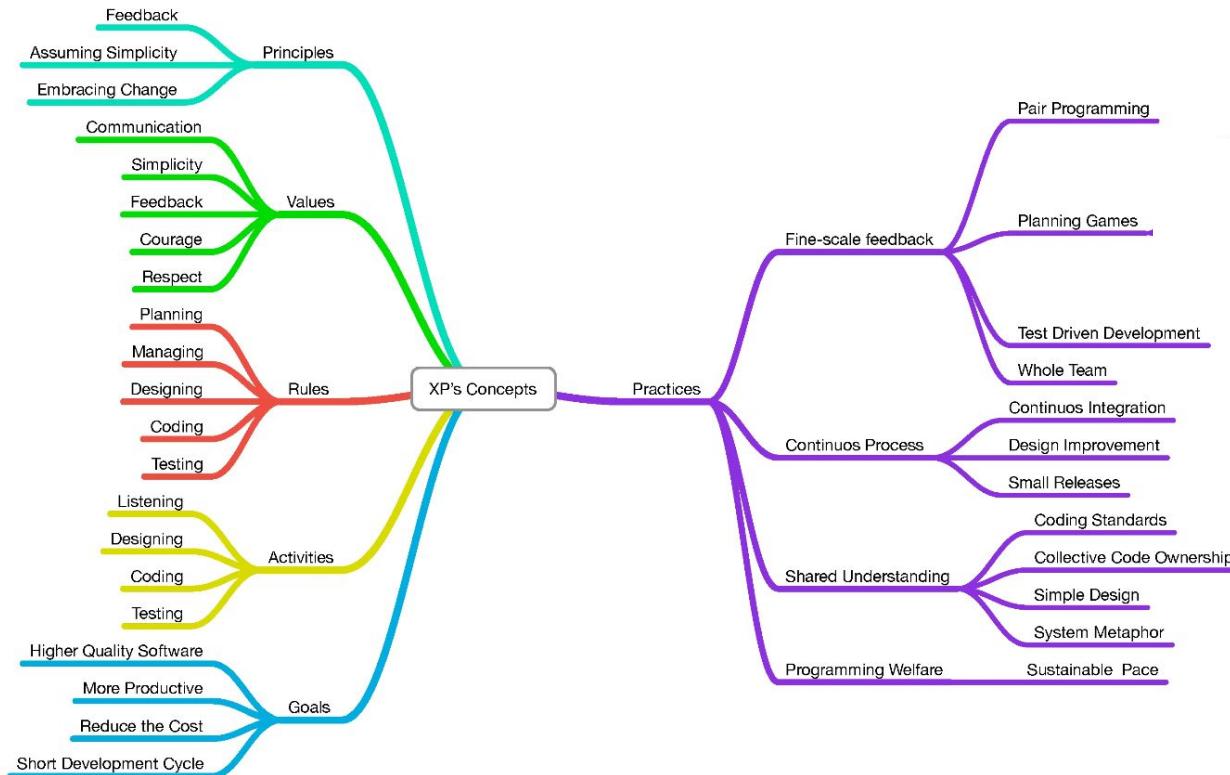
57,075 responses; select all that apply

<https://insights.stackoverflow.com/survey/2018/#development-practices>

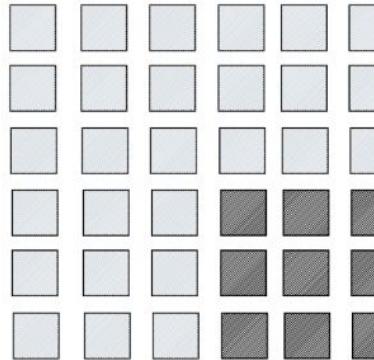
KANBAN

Backlog	Especificação		Implementação		Teste	
	Fazendo	Feito	Fazendo	Feito	Fazendo	Feito
	WIP História 2	Tarefa 6	Tarefa 4	Tarefa 3	Tarefa 2	Tarefa 1
			Tarefa 7	Tarefa 5		

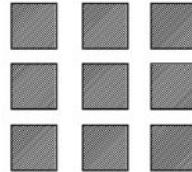
Extreme Programming (XP)



SCRUM CHEAT SHEET



PRODUCT
BACKLOG



SPRINT
BACKLOG



SPRINT



PSI

ROLES

ARTIFACTS

ACTIVITIES

PRODUCT OWNER

Develops product vision
Owns product backlog
Prioritizes backlog items

SCRUM MASTER

Process coach
Removes impediments
Facilitates team meetings

THE TEAM

Cross-functional
Self-organizing
Has skills to complete the sprint work

PRODUCT BACKLOG

Lists all the work on a product or project
Never complete, always changing
Higher order = higher priority

SPRINT BACKLOG

Lists work to be done in the current sprint
Pulled from product backlog
Items are broken into tasks

SHIPPABLE INCREMENT

Potentially shippable product increment
Completely designed, coded, and tested
Meets all acceptance criteria

SPRINT PLANNING

Plan created for what is to be delivered in the upcoming sprint.

THE SPRINT

Consistent duration of time where the team completes work.

DAILY SCRUM

15 minute meeting where team reports:
1.What did you do yesterday?
2.What are you doing today?
3.List any impediments.

SPRINT REVIEW

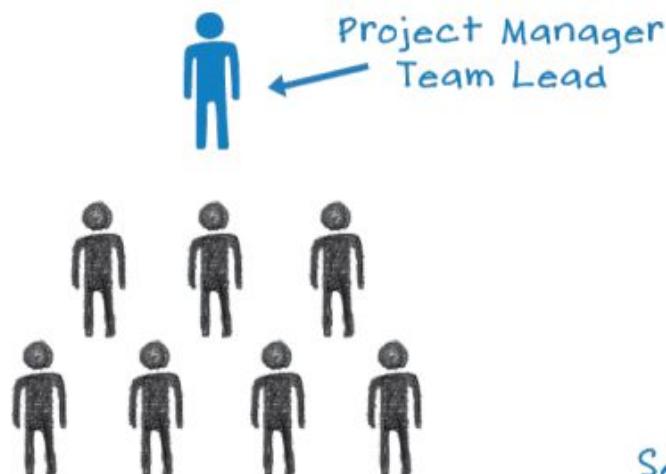
Team shows off completed work to Product Owner

SPRINT RETRO

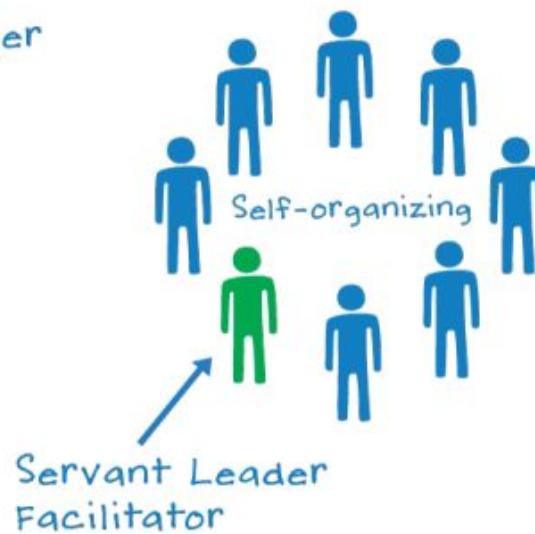
Team discusses how the sprint went and improvements to make next sprint

Papel do Scrum Master

Traditional Teams



Agile Teams



O papel do SCRUM MASTER



mantém a comunicação fluindo

mantém foco das reuniões

conhece o Scrum

agiliza comunicação com outras equipes (BDFTP, infra, etc)

resiste às adversidades

cobra horários do Customer Team

é comprometido

conhece os requisitos

escalpa problemas

mantém cronograma

reporta status

ScrumMaster / Team Facilitator
(role)



O que o Scrum Master NÃO deve ser:

passador de problema



— ¡Qué bonito! ¿Cómo se llama?

- Scrum Master
- ¿Puedo acariciarlo?
- Sí, claro, no hace nada.



Para o Product Owner

- 1) Claramente comunicar a visão, objetivo e itens do Backlog do Produto para a Equipe de Desenvolvimento (Não é proxy!!);
- 2) Compreender a longo-prazo o planejamento do Produto no ambiente empírico;
- 3) Compreender e praticar a agilidade;

Para a Equipe de Desenvolvimento

- 1) Remover impedimentos para o progresso da Equipe de Desenvolvimento;
- 2) Facilitar os eventos Scrum conforme exigidos ou necessários;



Ready? Prontos?

Antes de começar...

- a preparação do material foi adequada?
- a comunicação foi adequada?
- o timing foi adequado?



Conceito de READY (PRONTO)

PROBLEMAS:

- Reuniões de planning longas e improdutivas;
- Requisitos sendo definidos ao invés de somente expostos para a equipe.

CAUSA:

- Levantamento de requisitos incompleto ou inexistente.



Conceito de READY (PRONTO)

Para uma user story ser **iniciada** adequadamente, ela deve cumprir critérios de **Ready**





Conceito de **READY (PRONTO)**

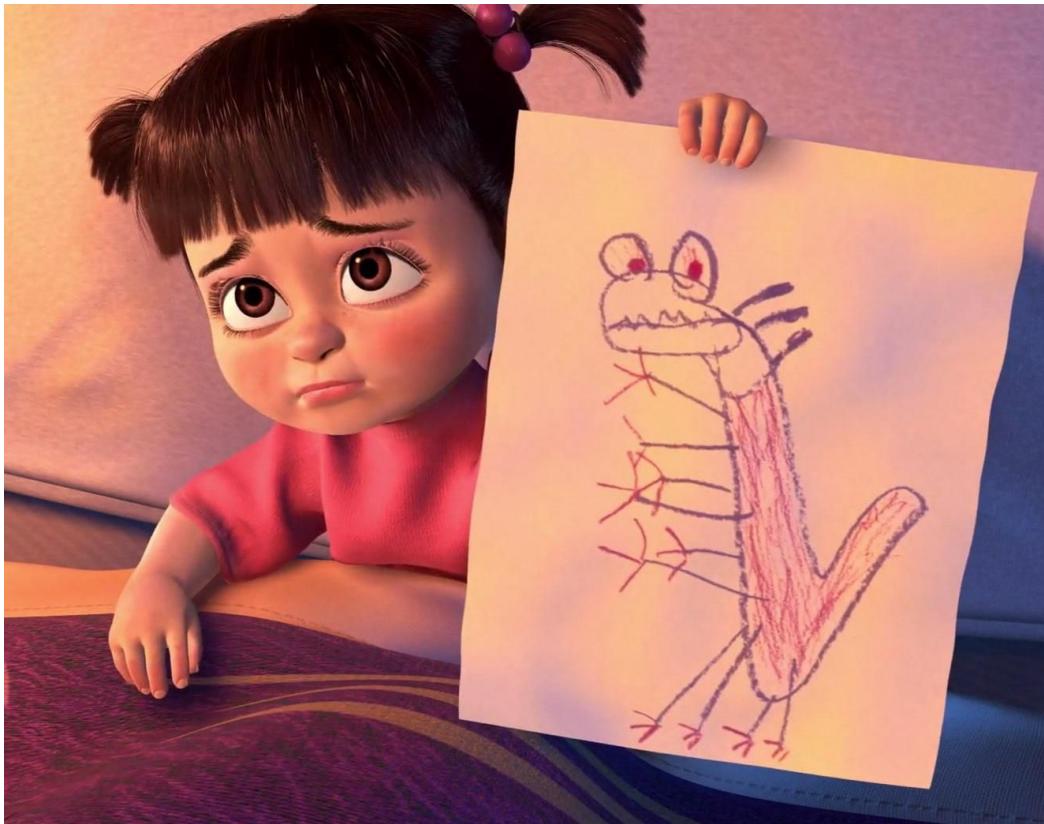
Na **entrega** da sprint, uma user story só é considerada entregue se atende os critérios de **DONE**.

A close-up photograph of a person's hands tying the laces of a blue and yellow running shoe. The person is wearing dark shorts and a white shirt. The background is blurred, suggesting an outdoor setting like a park.

Critérios de **READY (PRONTO)**

- User story definida
- Critérios de aceitação definidos
- Protótipos construídos

User Stories



User Stories

- Uma descrição curta e simples da funcionalidade, contada do ponto de vista de quem deseja a funcionalidade

The illustration shows a woman with blonde hair and a pink headband, wearing a purple shirt and teal pants, holding a baby in a blue blanket. Four speech bubbles are positioned around her, each containing a step of a user story template:

- 1 Defina seu usuário final**
Quem usará seu produto?
Como mãe,
- 2 Especifique o que eles querem.**
Que solução você está oferecendo?
Quero verificar meu bebê dormindo sem entrar no quarto dele,
- 3 Descreva o benefício**
O que seu usuário ganhará ao usar seu produto?
para que eu saiba que ele está seguro sem perturbá-lo.
- 4 Adicione critérios de aceitação**
O que determina que essa história seja "concluída"?
por exemplo, alerta a ser enviado ao smartphone registrado se um problema for detectado.

Copyright © 2018 Knowledge Train Limited

User Stories



Independent : Histórias devem ser independentes uma das outras;

Negotiable : Histórias não são contratos, mas lembretes para discussões;

Valuable : Histórias devem agregar valor para o cliente;

Estimatable : Os desenvolvedores devem ser capazes de estimar o tamanho das estórias;

Small : Histórias grandes dificultam as estimativas. Bem como histórias muito pequenas. Quebre ou agrupe dependendo do caso.

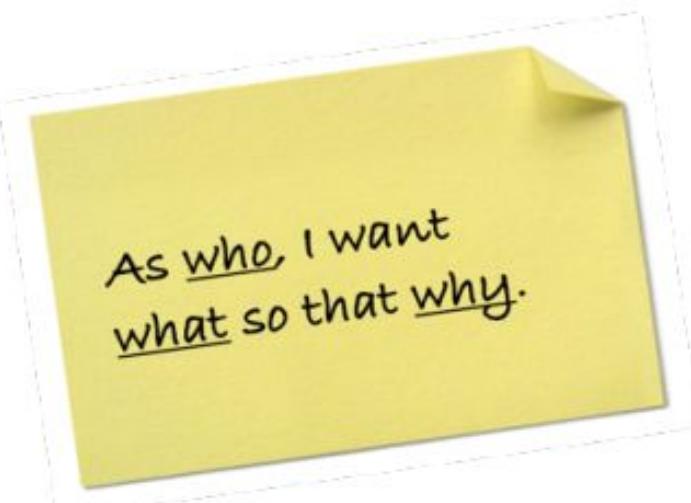
Testable : Histórias devem ser possíveis de serem testadas.

EU COMO <usuário, papel, pessoa>

DESEJO <fazer alguma coisa>

DE MODO QUE <eu atinja um objetivo de negócio>

- Uma ação sendo realizada
- Um papel que realiza esta ação
- O benefício que esta ação traz



As who, I want
what so that why.

Critérios de Aceitação / Acceptance Criteria



“Condições que um produto de software deve satisfazer para que seja aceita por um usuário, cliente ou algum outro stakeholder.”

Uma user story indica o objetivo de negócio a ser atingido. Os critérios de aceitação indicam as condições para que este objetivo seja alcançado.

Critérios de Aceitação / Acceptance Criteria

Características

- Sentenças com descrições claras sobre a aceitação do sistema.
- Devem ser expressas de maneira clara e simples, em linguagem ubíqua
- Podem ser funcionais ou não-funcionais
- Sem ambiguidade, deve indicar o que é ou não é aceitável
- Não existe aceite parcial: o critério é atendido ou não
- Se bem escritos, deverão ser facilmente traduzidos em casos de testes.



Critérios de Aceitação / Acceptance Criteria

Exemplos

“Passageiro não pode escolher um assento já ocupado.”

“Comprador deve selecionar uma forma de pagamento”

“O tempo de resposta não deve ser superior a 15 segundos”

Critérios de Aceitação / Acceptance Criteria

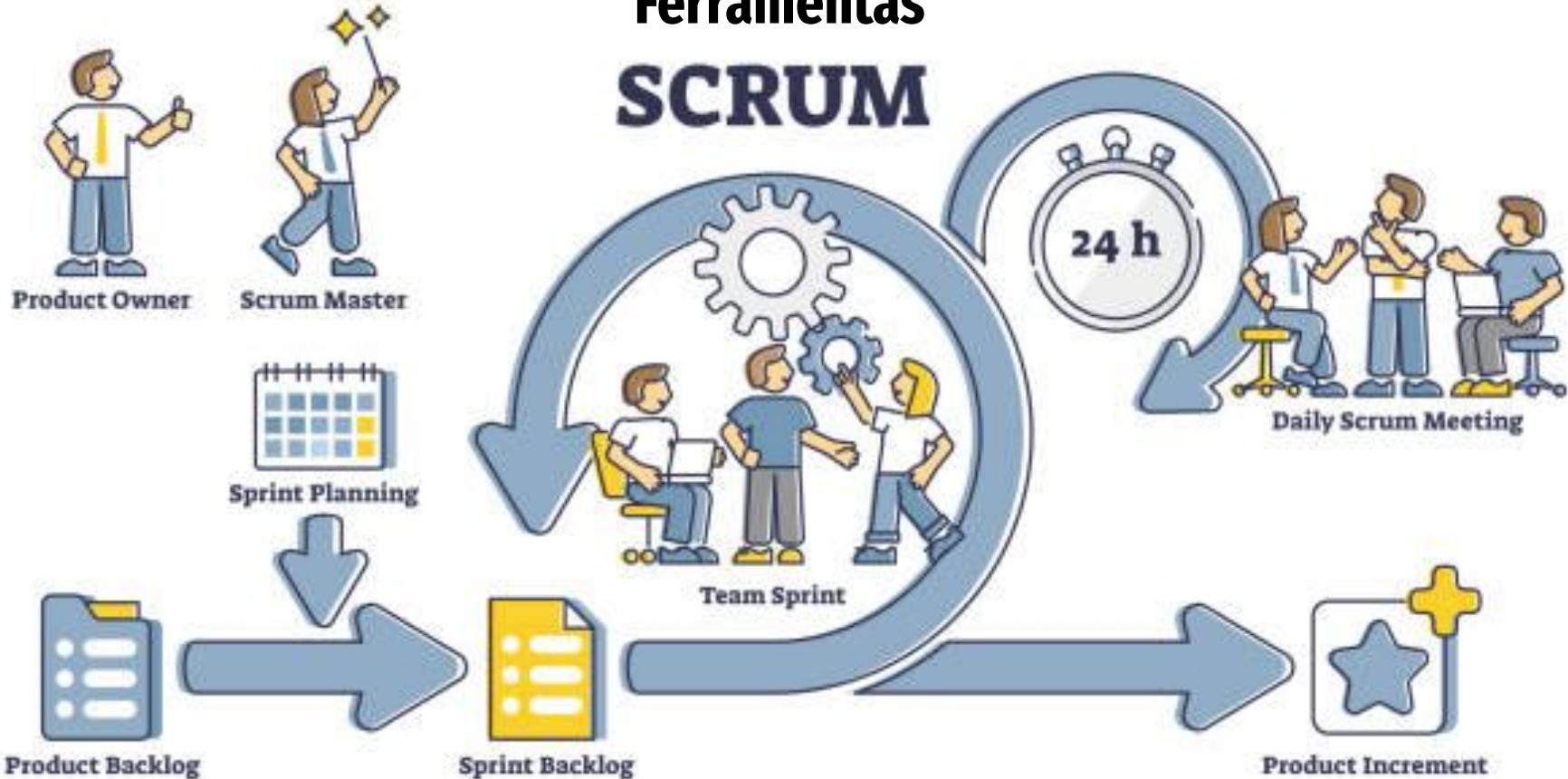
Critério não é solução

Critérios de aceitação devem ser independentes de implementação. Em geral, critérios devem indicar uma intenção, e não como uma ação deverá acontecer.

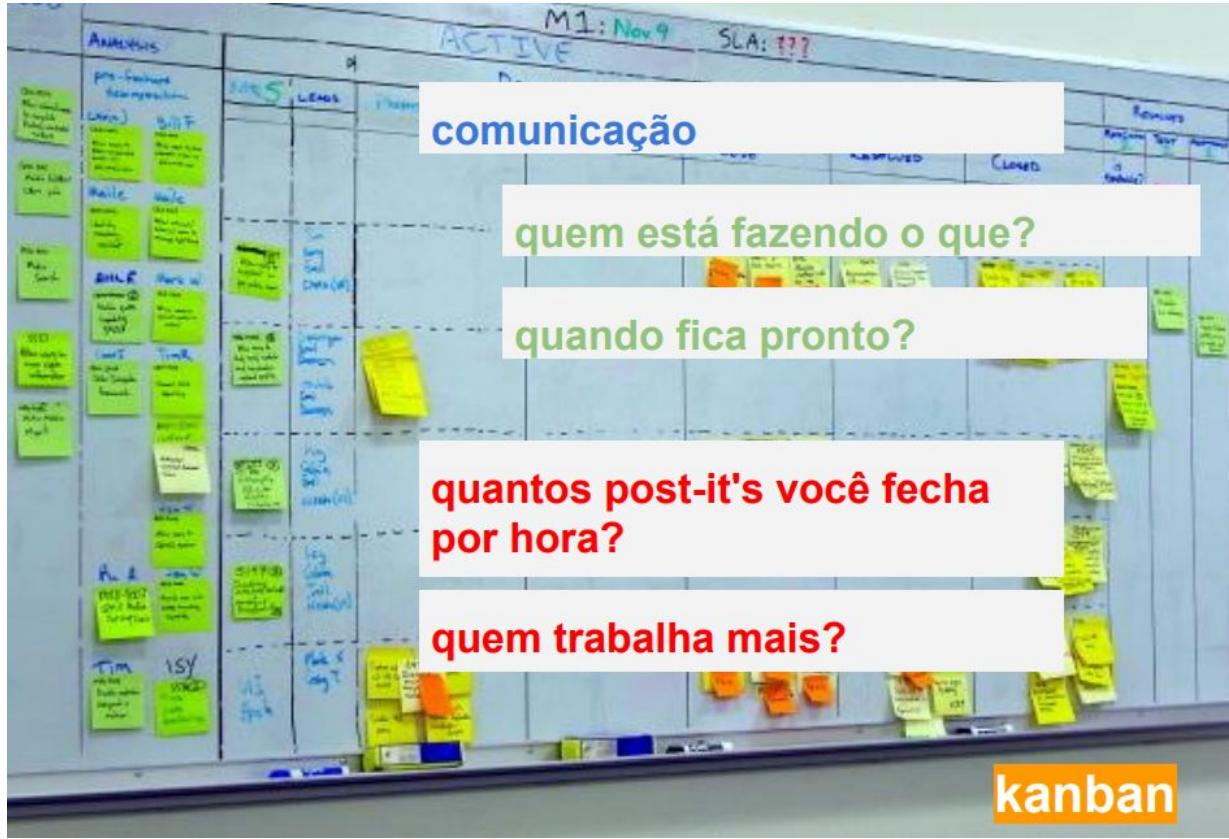
ex: “Quando o usuário salvar, a operação é efetuada, e o sistema indica sucesso”

Ferramentas

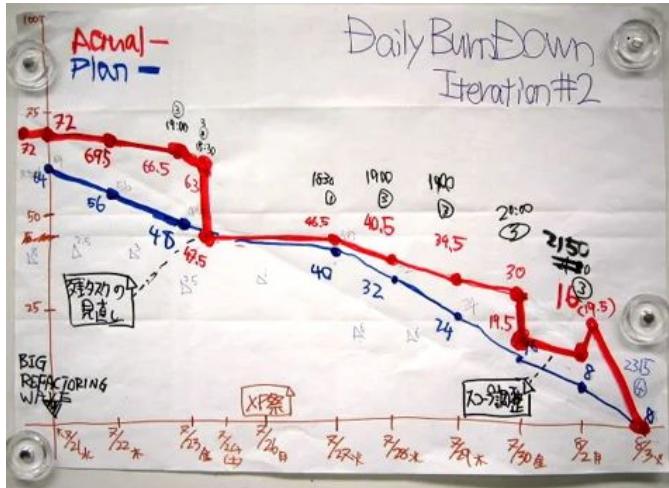
SCRUM



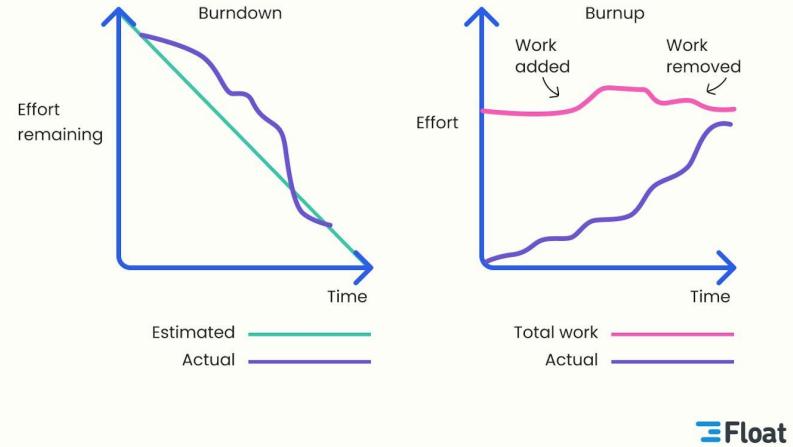
Ferramentas - Kanban



Ferramentas - Burn[Up|Down] chart



Burndown vs. Burnup charts



COMUNICAÇÃO
Estamos dentro do planejado?
Tem algo nos impedindo?

Ferramentas - Stand Up (Daily) Meeting

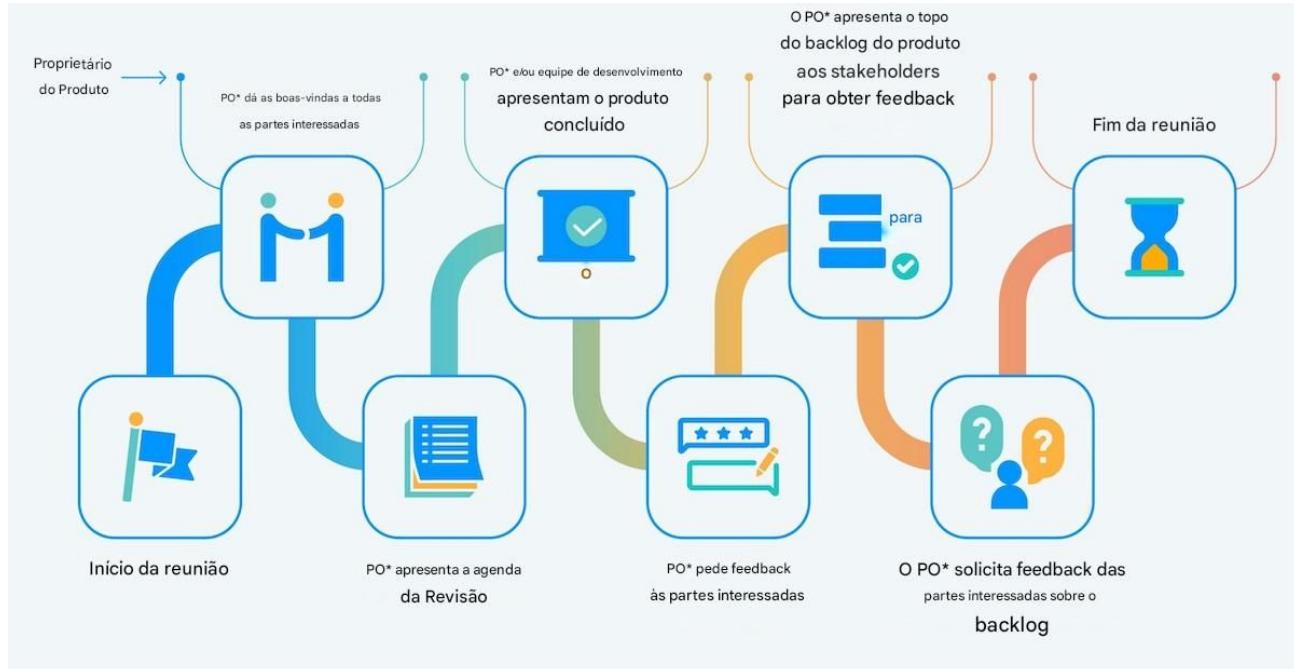
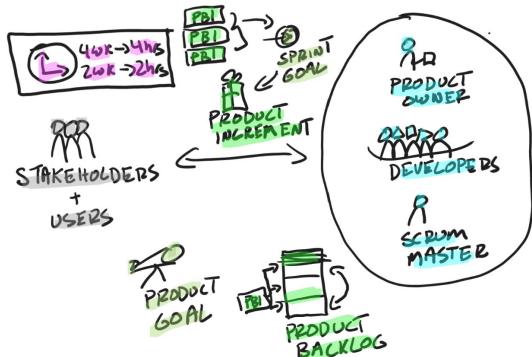
COMUNICAÇÃO

O que eu fiz que ajudou a atingir
nossa meta?

O que farei para ajudar a atingir
nossa meta?

Há algum impedimento que possa
atingir nossa meta?

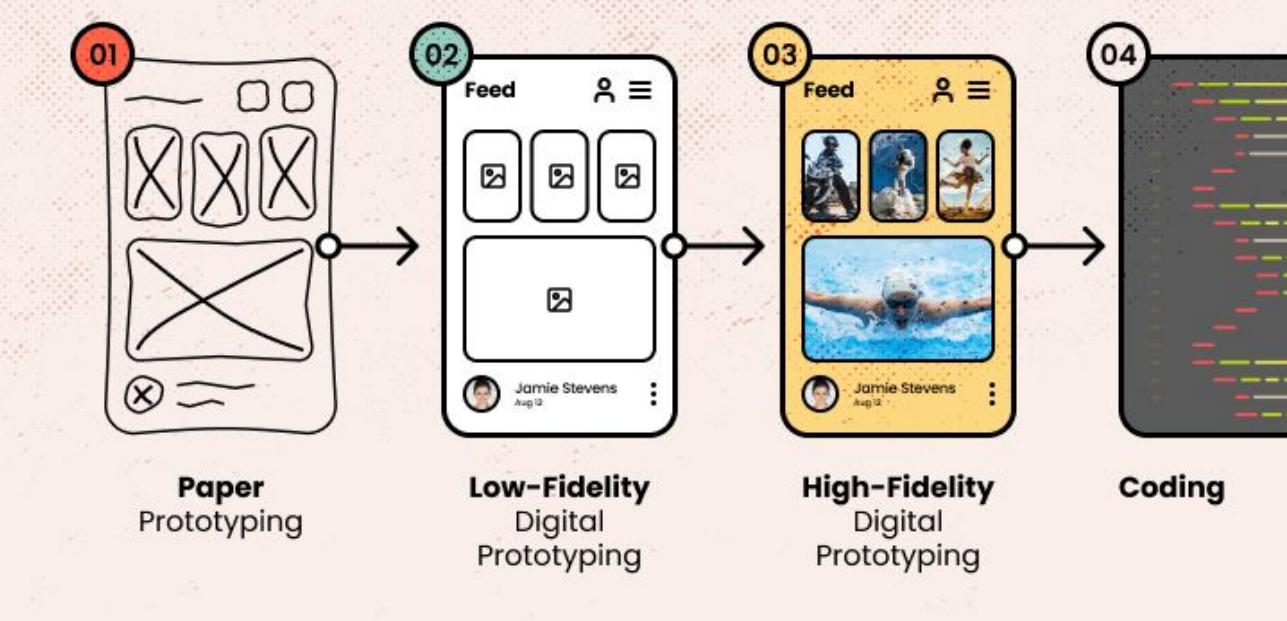
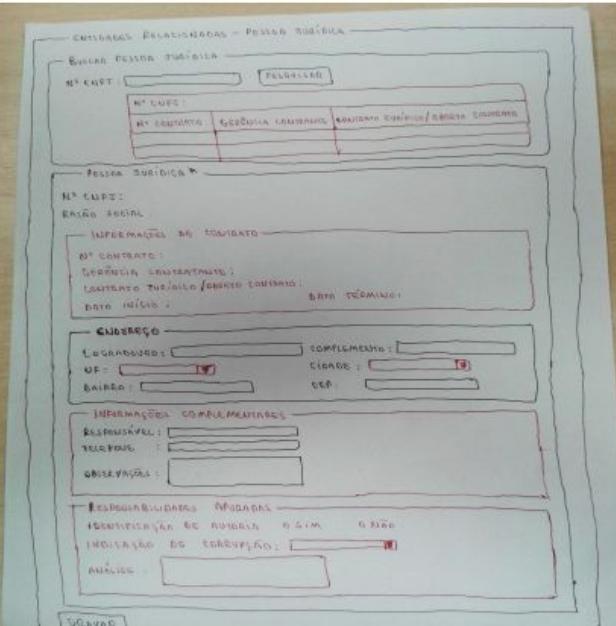
Ferramentas (Atividades) - Sprint Review

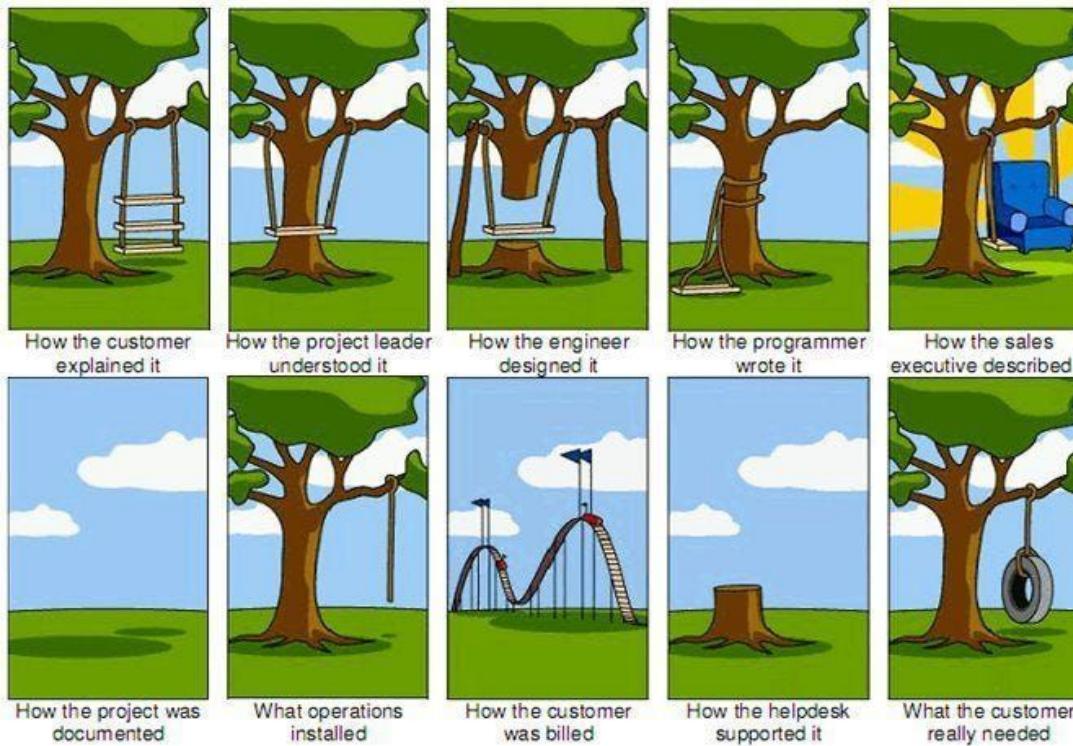


Ferramentas (Atividades) - Sprint Retrospective



Ferramentas - Protótipos





USER STORY: O que eu quero fazer?

PROTÓTIPO: Como isso deve parecer?

CRITÉRIOS DE ACEITAÇÃO (Ready): A história está completa?

CRITÉRIOS DE ACEITAÇÃO (Done): O software faz o que foi pedido?

RECAPITULANDO

CARTÃO

É onde as user stories são escritas.

Pequeno para caber apenas o essencial.

Texto suficiente para lembrar do que está sendo desenvolvido.

Conversa

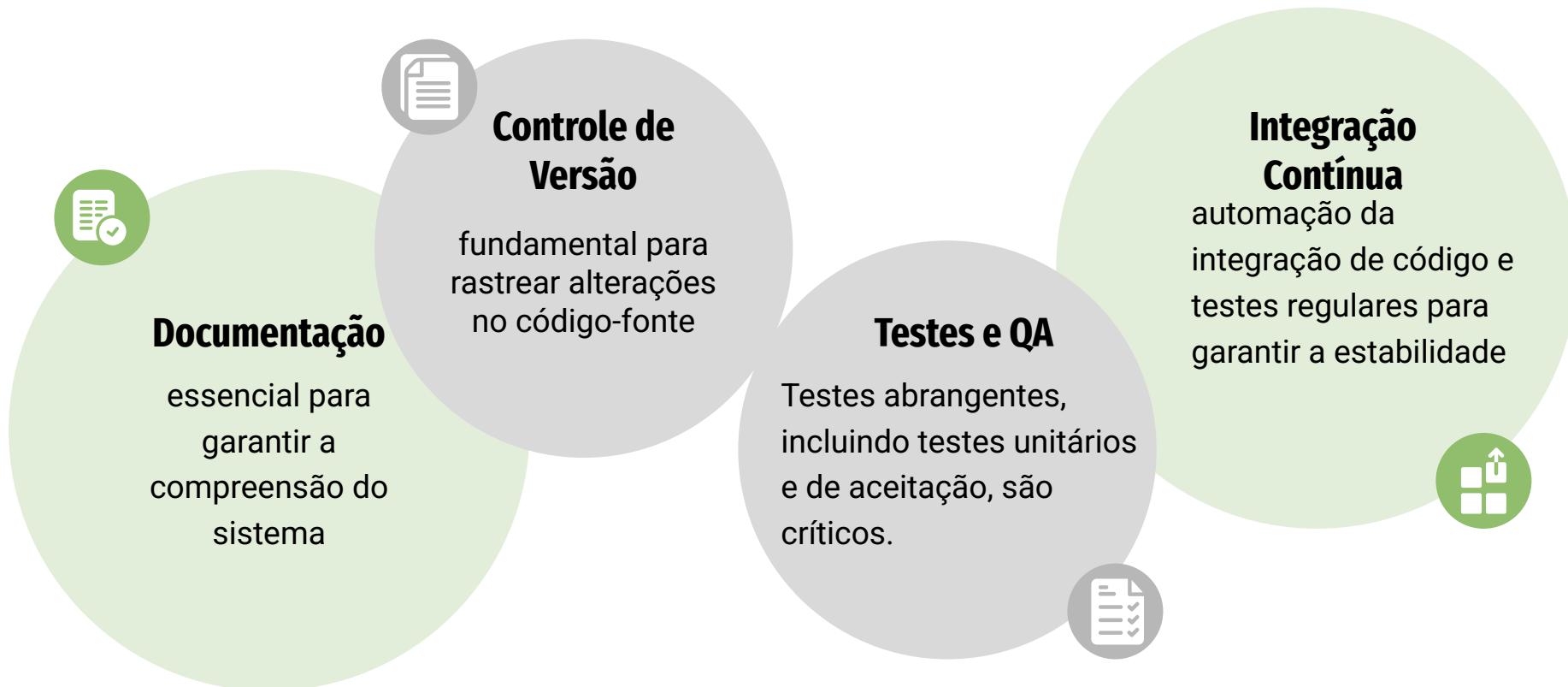
Documentação da regra de negócio acontece principalmente verbalmente entre PO e equipe.
As características e idéias levantadas são registradas como critérios de aceitação.

Confirmação

No início da sprint o PO explica e prioriza pra equipe o que ele precisa e como ele vai confirmar que o sistema atende a necessidade.



Boas Práticas



Atividade

Selecione um software de seu interesse e busque informações sobre seu processo de desenvolvimento:

- Acesse a página ou manual do software escolhido.
- Se for um software livre, localize seu repositório.
- Existe documentação disponível sobre seu código fonte?
- Como e em que periodicidade são publicadas as versões (releases) do software?
- Como são reportados os bugs e sugestões de melhorias?
- Qual metodologia de desenvolvimento adotada?
- Você concorda com a metodologia utilizada? Qual outra ou outras metodologias poderiam ser adotadas?

Reúna o máximo de informações sobre o processo de desenvolvimento do software escolhido e poste até a próxima aula seu relatório no Padlet da Turma.

Obrigado!

Qual sua impressão sobre a aula de hoje ?

