CTEDS

Capacitação Tecnológica em Engenharia e Desenvolvimento de Software

Métodos Ágeis

Introdução

Professor: Carla Rocha [UnB] *caguiar@unb.br*

carlarocha.org

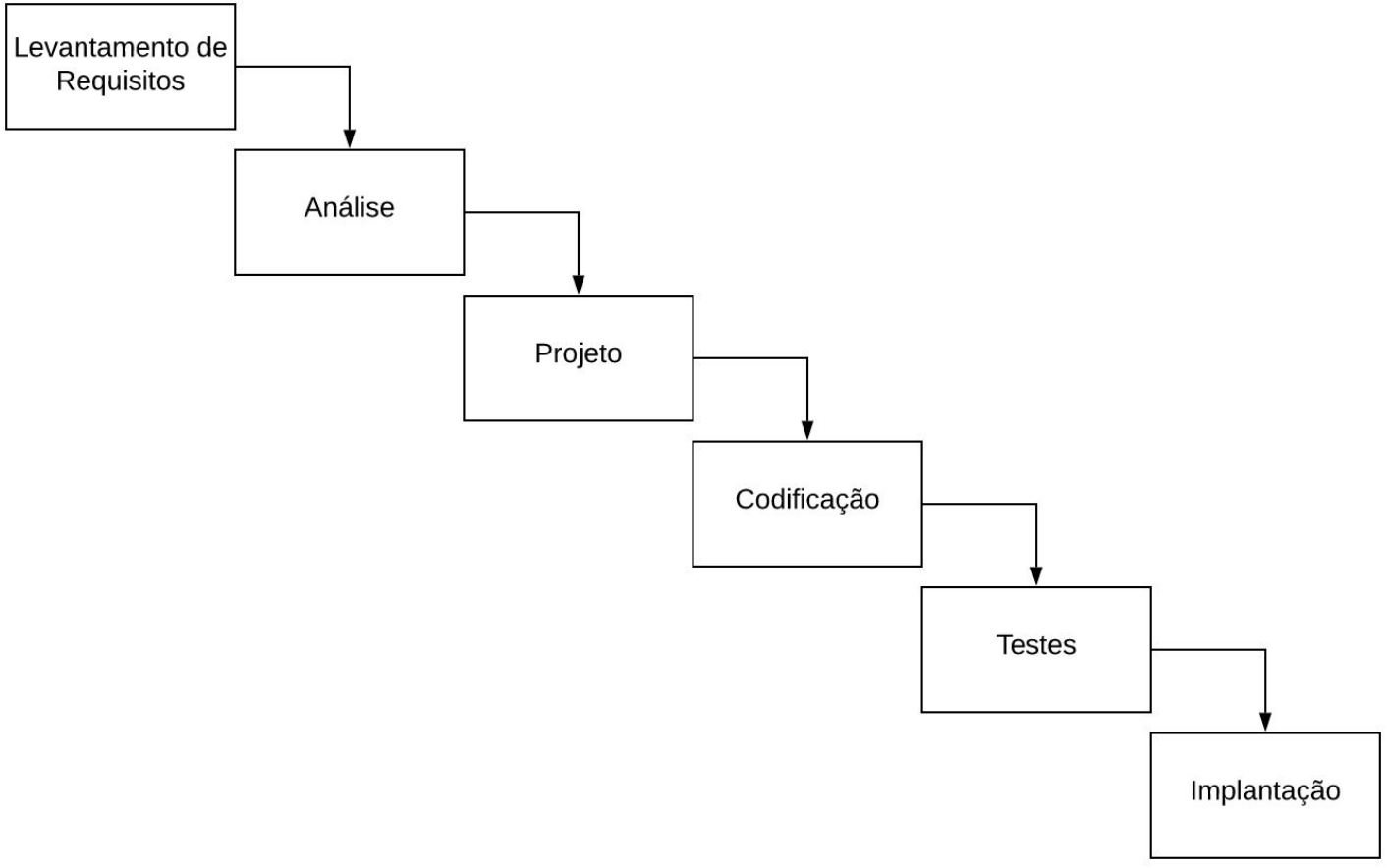
Agenda

- Engenharia Tradicional - Modelo Cascata
- Manifesto Ágil
- Extreme Programming
- Scrum
- A disciplina
  - Avaliação/Projeto
  - Onboarding
  - Git

Engenharia Tradicional

- Civil , mecânica, elétrica, aviação, automobilística, etc
- Projeto com duas características:
  - Planejamento detalhado (big upfront design)
  - Sequencial
- Isto é: Waterfall
  - Há milhares de anos

Natural que ES começasse usando Waterfall



Waterfall não funcionou com Software!

Software é Diferente

- Engenharia de Software ≠ Engenharia Tradicional
- Software ≠ (carro, ponte, casa, avião, celular, etc)
- Software ≠ (produtos físicos)
- Software é abstrato e "adaptável"

Dificuldade 1: Requisitos

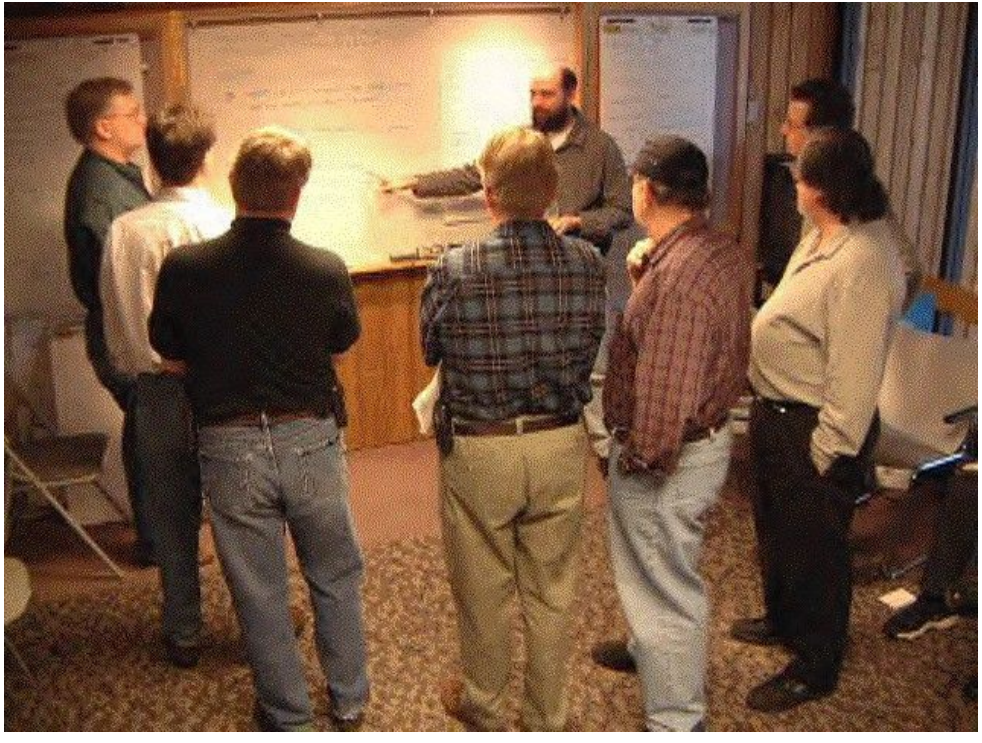
- Clientes não sabem o que querem (em um software)
  - Funcionalidades são "infinitas" (difícil prever)
  - Mundo muda!
- Não dá mais para ficar 1 ano levantando requisitos, 1 ano projetando, 1 ano implementando, etc
- Quando o software ficar pronto,

  ele estará obsoleto!
-

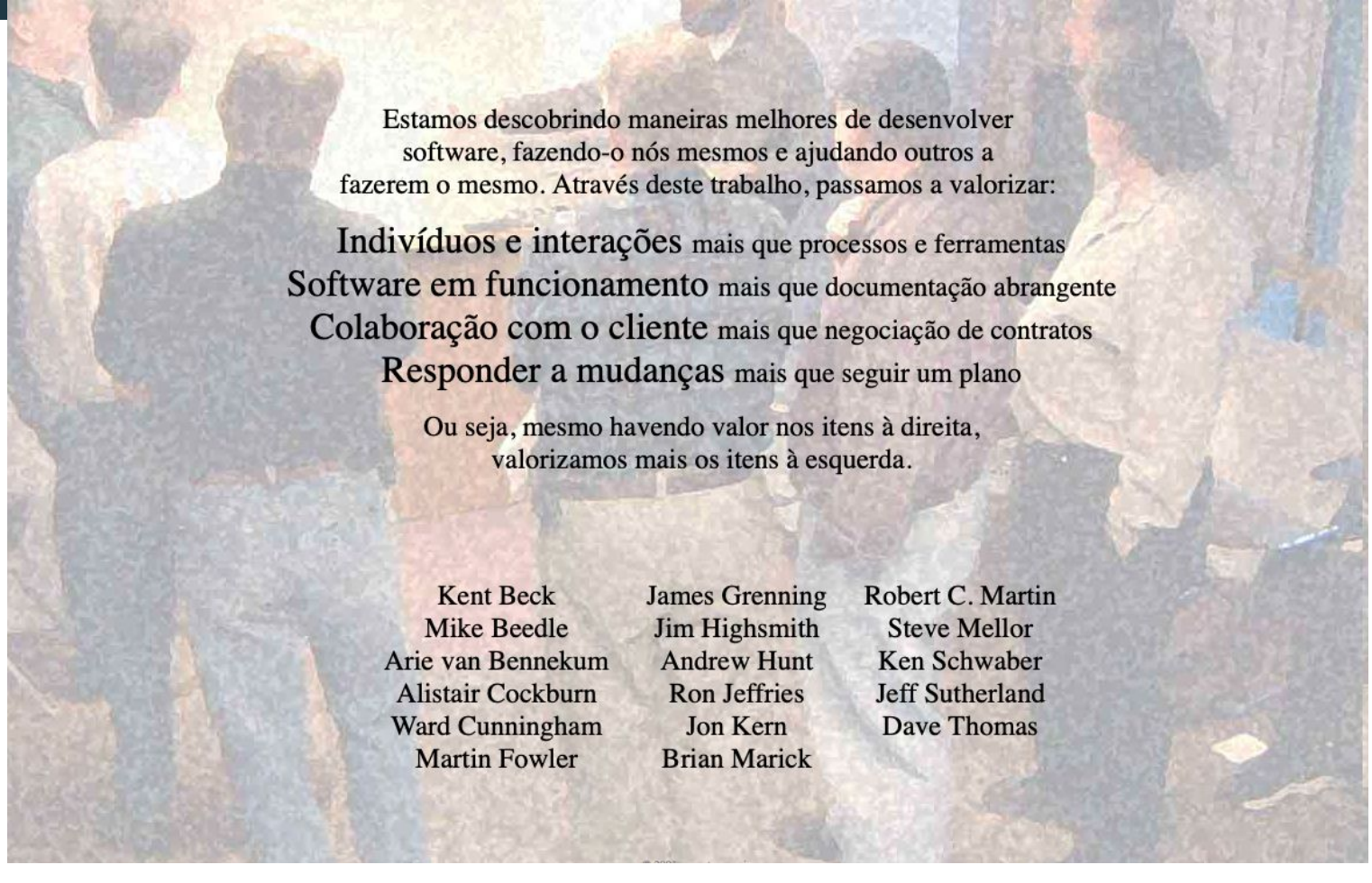
Dificuldade 2: Documentação Detalhadas

- Verbosas e pouco úteis
- Na prática, desconsideradas durante implementação
- *Plan-and-document* não funcionou com software

Manifesto Ágil



Manifesto Ágil



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

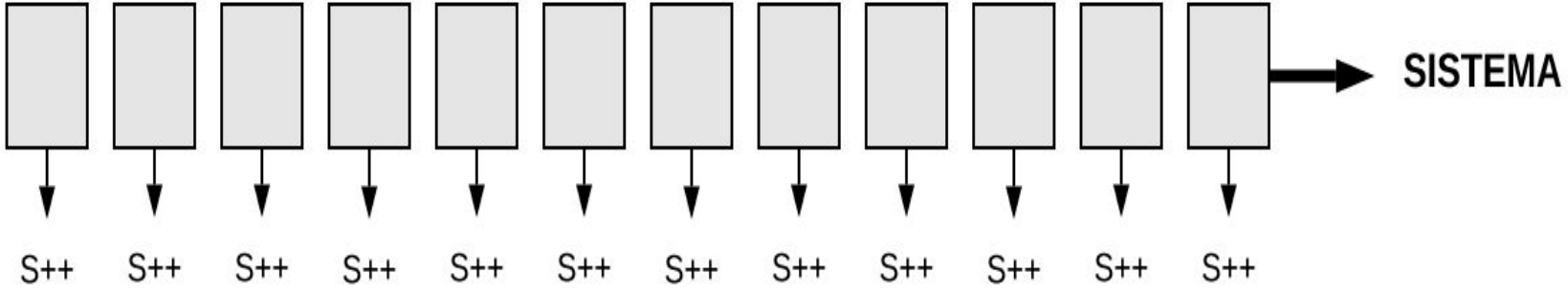
Ou seja, mesmo havendo valor nos itens à direita,
valorizamos mais os itens à esquerda.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

Ideia Central: Desenvolvimento Iterativo

Waterfall

Ágil



Desenvolvimento Iterativo

- Suponha um sistema imenso, complexo etc
- Qual o menor "incremento de sistema" eu consigo implementar em 15 dias e validar com o usuário?
- Validar é muito importante!
- Cliente não sabe o que quer!

Reforçando:
Ágil = Iterativo

Outros Pontos Importantes

- Menor ênfase em documentação
- Menor ênfase em *big upfront design*
- Envolvimento constante do cliente

Outros Pontos Importantes

- Novas práticas de programação
  - Testes, refactoring, integração contínua, etc

Métodos Ágeis

Métodos Ágeis

- Dão mais consistência às ideias ágeis
  - Definem um processo, mesmo que leve
  - Workflow, eventos, papeis, práticas, princípios etc

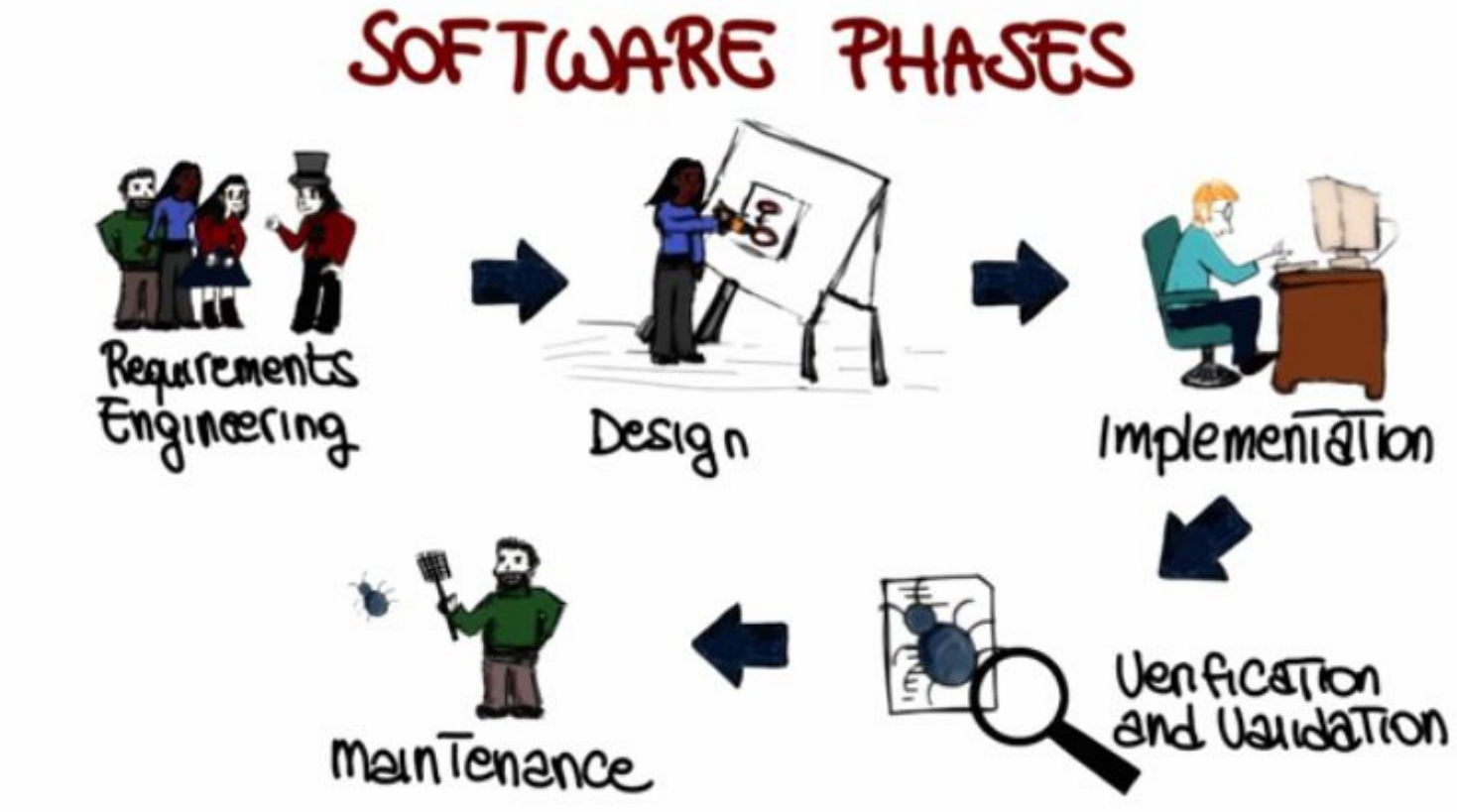
Métodos Ágeis que Vamos Estudar

- Extreme Programming (XP)
- Scrum
- Kanban

Antes de Começar

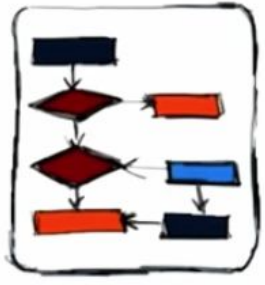
- Nenhum processo é uma bala-de-prata
- Processo ajuda a não cometer certos "grandes erros"
- Processos não são adotados 100% igual ao manual
  - Bom senso é importante
  - Experimentação é importante

Antes de Começar



Antes de Começar

Methodologies



Techniques

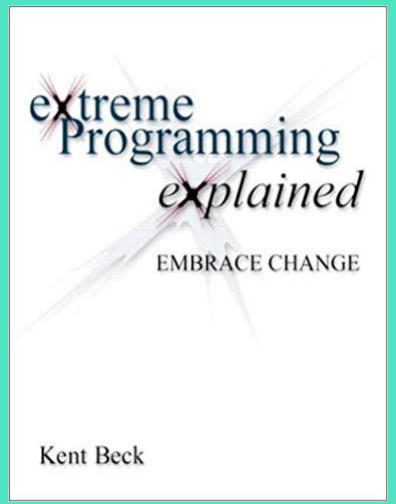
Tools

High Quality software that Works and Fits Budget

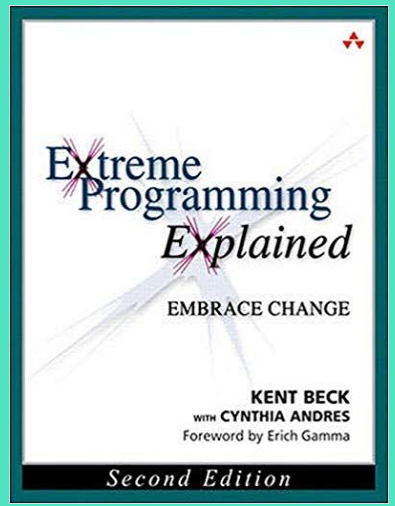
Extreme Programming (XP)

Extreme Programming

Kent Beck



1999



2004

XP

XP = Valores + Princípios + Práticas

Valores

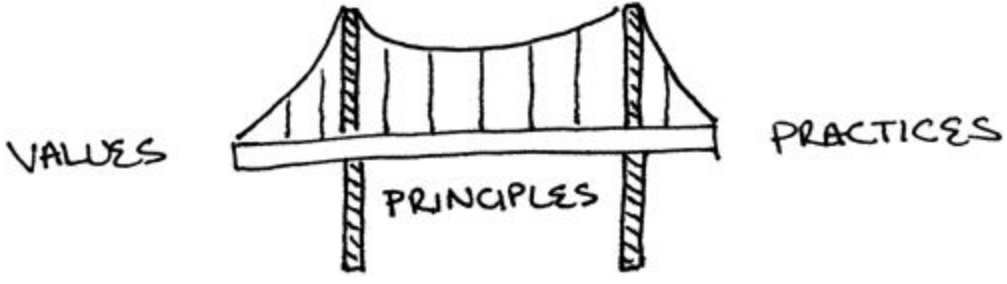
- Comunicação
- Simplicidade
- Feedback
- Coragem
- Respeito
- Qualidade de Vida (semana 40 hrs)

XP

Valores ou "cultura" são fundamentais em software!

XP

XP = Valores + Princípios + Práticas



Princípios

- Economicidade
- Melhorias Contínuas
- Falhas Acontecem
- Baby Steps
- Responsabilidade Pessoal

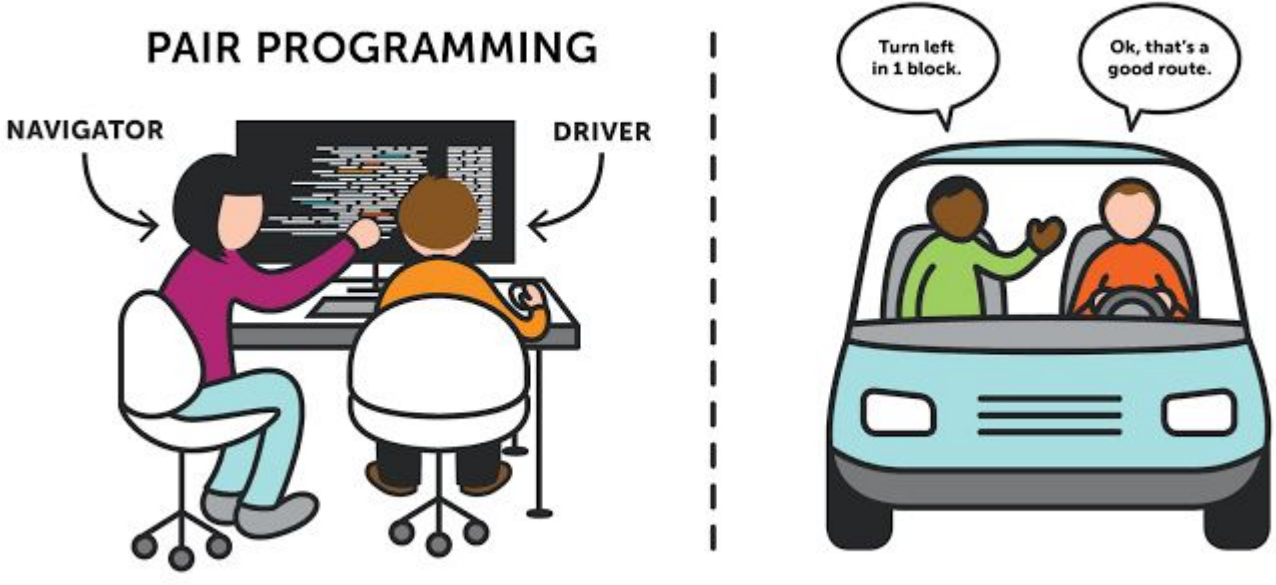
XP

XP = Valores + Princípios + Práticas

Práticas

Práticas sobre o Processo de Desenvolvimento	Práticas de Programação	Práticas de Gerenciamento de Projetos
Representante dos Clientes Histórias de Usuário Iterações Releases Planejamento de Releases Planejamento de Iterações Planning Poker Slack	Design Incremental Programação Pareada Testes Automatizados Desenvolvimento Dirigido por Testes (TDD) Build Automatizado Integração Contínua	Ambiente de Trabalho Contratos com Escopo Aberto Métricas

Programação Pareada



Estudo com Engenheiros da Microsoft (2008)

- Vantagens:
  - Redução de bugs
  - Código de melhor qualidade
  - Disseminação de conhecimento
  - Aprendizado com os pares
- Desvantagem:
  - Custo

Contrato com Escopo Aberto

- Escopo fechado
  - Cliente define requisitos ("fecha escopo")
  - Empresa desenvolvedora: preço + prazo
- Escopo aberto
  - Escopo definido a cada iteração
  - Pagamento por homem/hora
  - Contrato renovado a cada iteração

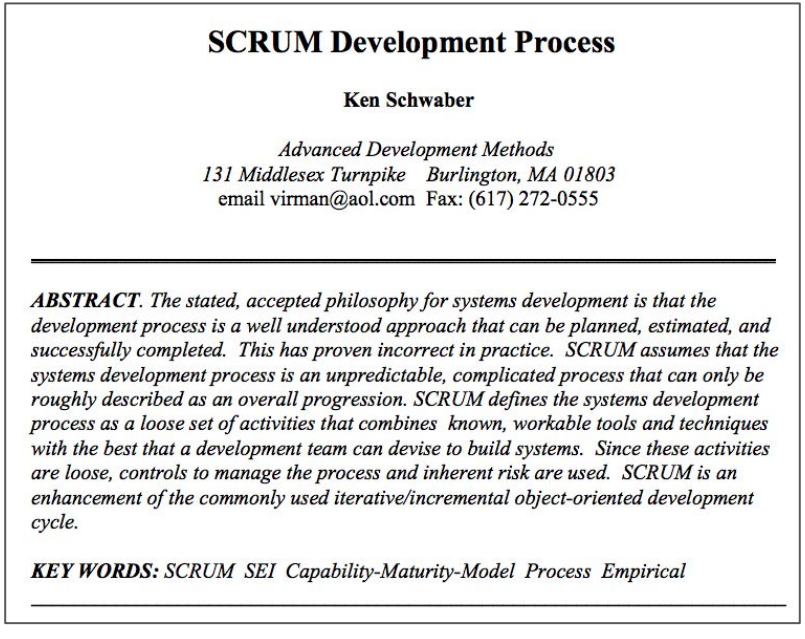
Contrato com Escopo Aberto

- Exige maturidade e acompanhamento do cliente
- Vantagens:
  - Privilegia qualidade
  - Não vai ser "enganado" ("entregar por entregar")
  - Pode mudar de fornecedor

Scrum

Scrum

- Proposto por Jeffrey Sutherland e Ken Schwaber (OOPSLA 1995)



SCRUM Development Process

Ken Schwaber

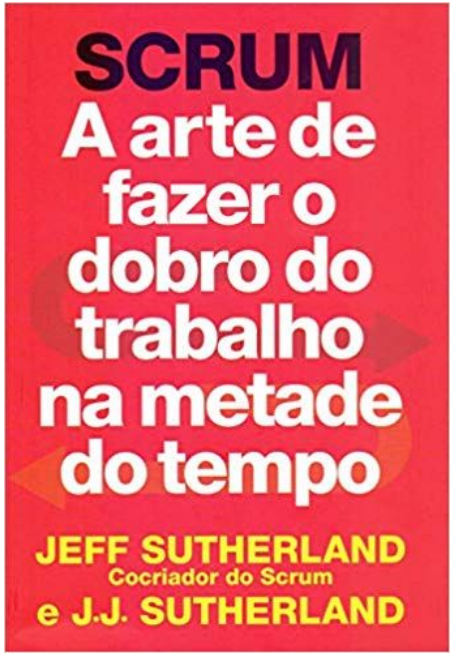
Advanced Development Methods
131 Middlesex Turnpike Burlington, MA 01803
email virman@aol.com Fax: (617) 272-0555

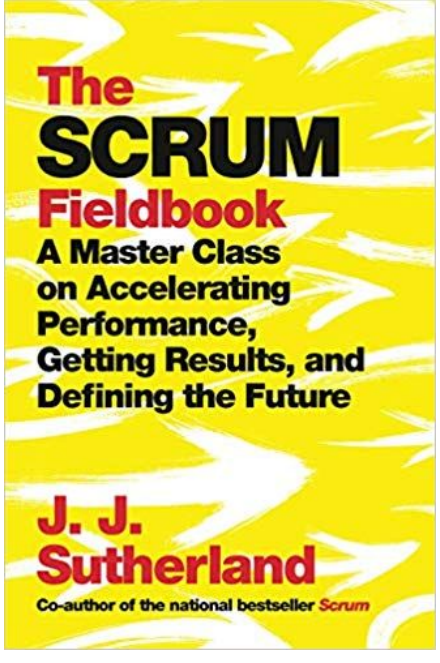
ABSTRACT. *The stated, accepted philosophy for systems development is that the development process is a well understood approach that can be planned, estimated, and successfully completed. This has proven incorrect in practice. SCRUM assumes that the systems development process is an unpredictable, complicated process that can only be roughly described as an overall progression. SCRUM defines the systems development process as a loose set of activities that combines known, workable tools and techniques with the best that a development team can devise to build systems. Since these activities are loose, controls to manage the process and inherent risk are used. SCRUM is an enhancement of the commonly used iterative/incremental object-oriented development cycle.*

KEY WORDS: *SCRUM SEI Capability-Maturity-Model Process Empirical*

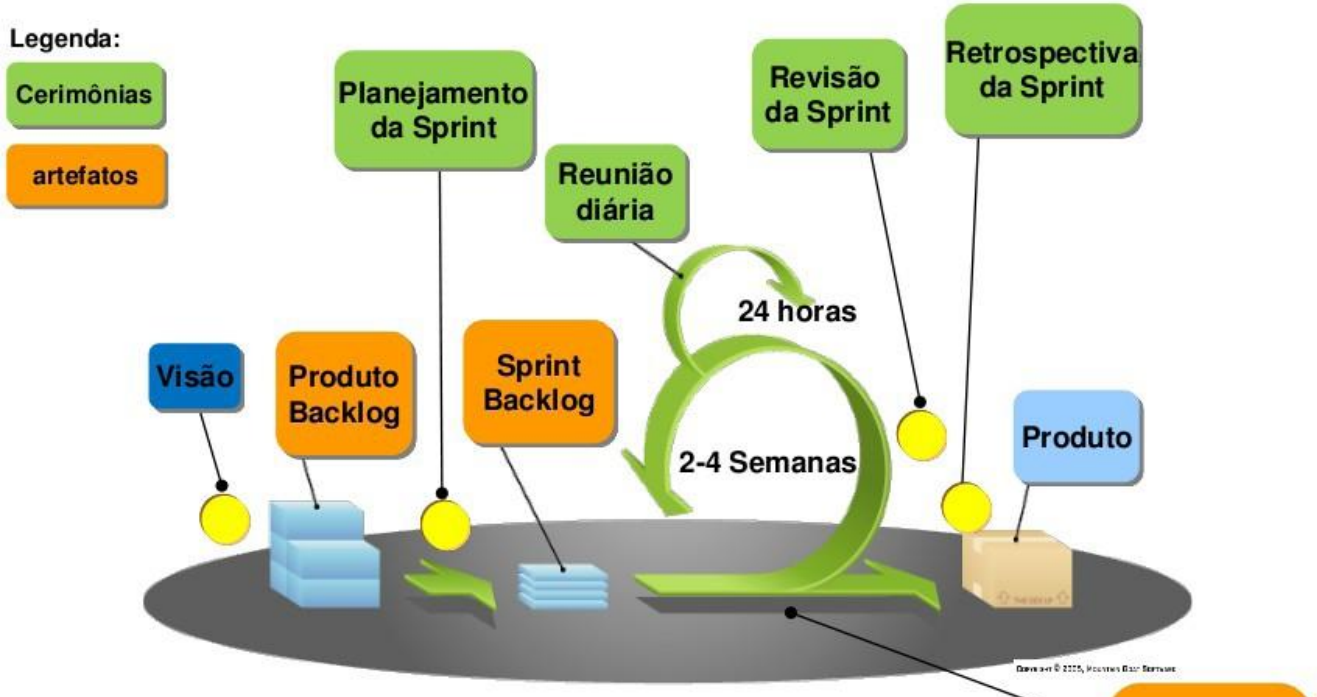
Scrum

- Scrum é uma indústria: livros, consultoria, certificações, marketing





Scrum

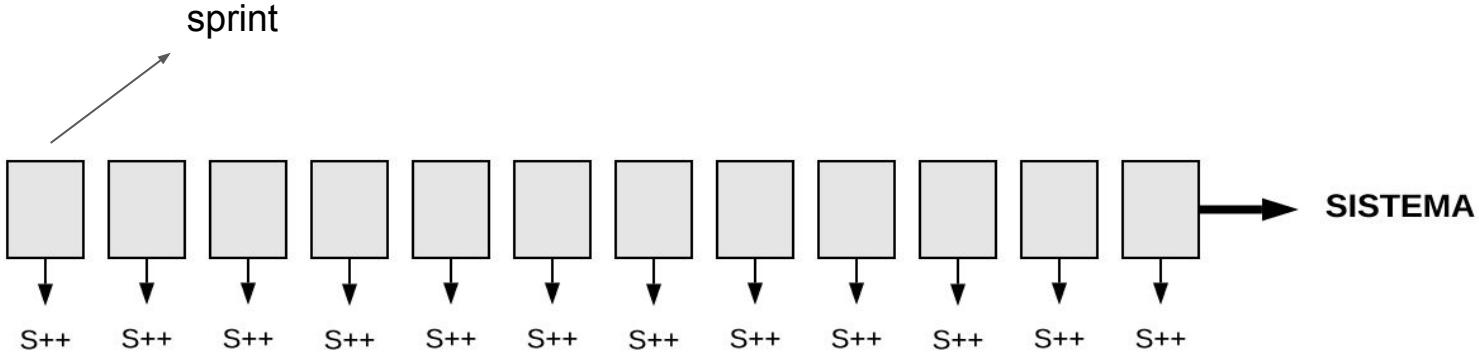

Legenda:
Cerimônias
artefatos
Planejamento da Sprint
Revisão da Sprint
Retrospectiva da Sprint
Reunião diária
24 horas
2-4 Semanas
Visão
Produto Backlog
Sprint Backlog
Produto
Papéis
• Product Owner (PO)
• ScrumMaster (SM)
• Equipe Scrum
Cerimônias
• Planejamento da Sprint
• Reunião Diária
• Revisão da Sprint
• Retrospectiva da Sprint
Artefatos
• Product Backlog
• Sprint Backlog
• Burndown (gráfico)
Burndown

Scrum vs XP

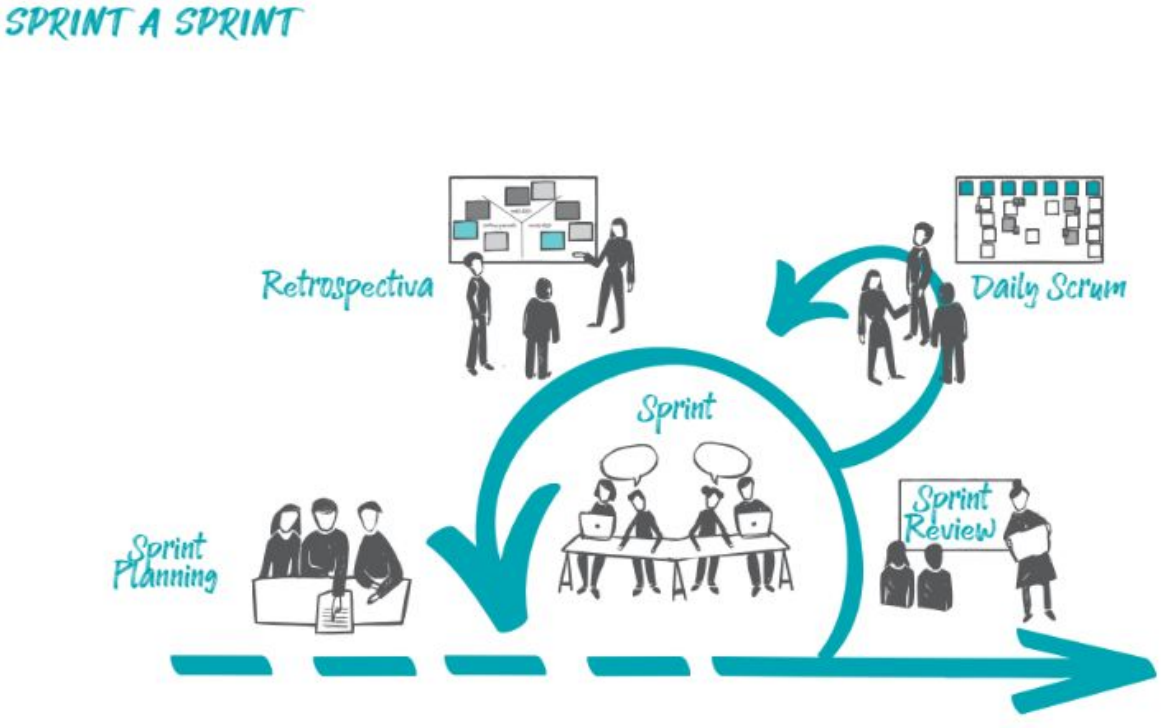
- Scrum não é apenas para projetos de software
  - Logo, não define práticas de programação, como XP
- Scrum define um "processo" mais rígido que XP
  - Eventos, papéis e artefatos bem claros

Principal Evento: Sprints

- Como em qualquer método ágil, desenvolvimento é dividido em sprints (iterações)
- Duração de um sprint: até 1 mês, normalmente 15 dias

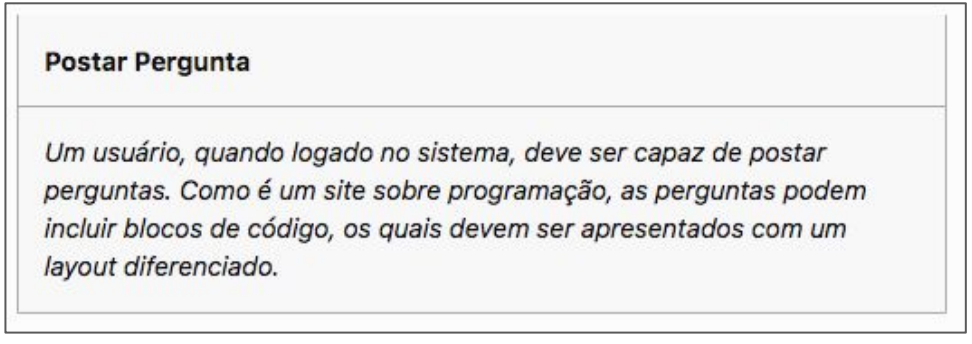


Principal Evento: Sprints



O que se faz em uma sprint?

- Implementa-se algumas **histórias dos usuários**
- Histórias = funcionalidades (ou features) do sistema
- Exemplo: fórum de perguntas e respostas



Bem simples, deve caber em um post-it

Quem escreve as Histórias

- **Product Owner (PO)**
- Membro (papel) obrigatório em times Scrum
- Especialista no domínio do sistema

Antes

Stakeholders

Analista de Requisitos

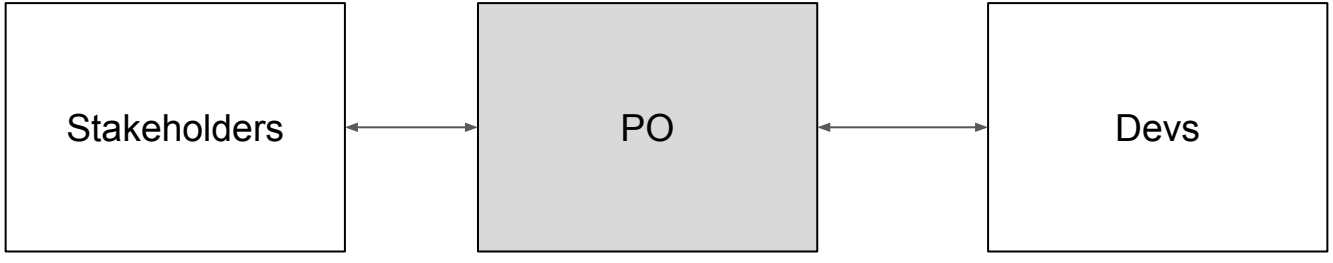
Devs

Linguagem natural
(poderia levar anos para ficar pronto)

Observação: stakeholders são os clientes do sistema e qualquer outra parte interessada ou afetada por ele. Exemplo: departamento jurídico é interessado no módulo de contratos de um sistema

Hoje (Scrum)

- Durante sprint, PO explica histórias (requisitos) para devs
- Troca-se documentação formal e escrita por documentação verbal e informal
- Conversas entre PO e Devs



Funções do PO

- Escrever histórias dos usuários
- Explicar histórias para os devs, durante o sprint
- Definir "testes de aceitação" de histórias
- Priorizar histórias
- Manter o backlog do produto

Backlog do Produto

- Lista de histórias do usuário, que foram escritas pelo PO
- Duas características:
  - Priorizada: histórias do topo têm maior prioridade
  - Dinâmica: histórias podem sair e entrar, à medida que o sistema evolui

Resumindo

- Sprint (evento)
- PO e Devs (papeis)
- Backlog do produto (artefato)

Quais histórias vão entrar no próximo sprint?

- Essa decisão é tomada no início do sprint
- Em uma reunião chamada de **planejamento do sprint**
- PO propõe histórias que gostaria de ver implementadas
- Devs decidem se têm **velocidade** para implementá-las

Importante!

- Em um time scrum, todos têm o mesmo nível hierárquico
- PO não é o "chefe" dos Devs
- Devs têm autonomia para dizer que não vão conseguir implementar tudo que o PO quer em um único sprint

Planejamento da Sprint

- 1a parte da reunião:
  - Definem-se as histórias do sprint
- 2a parte da reunião:
  - Histórias são quebradas em tarefas
  - Tarefas são alocadas a devs

Planejamento da Sprint - Exemplo

- História: Postar Perguntas
- Tarefas:
  - Projetar e testar a interface Web, incluindo layout, CSS templates, etc.
  - Instalar banco de dados, projetar e criar tabelas.
  - Implementar a camada de acesso a dados.
  - Implementar camada de controle, com operações para cadastrar, remover e atualizar perguntas.
  - Implementar interface Web.

Backlog da Sprint

- Lista de tarefas do sprint, com responsáveis e duração estimada

Sprint está pronta para começar!

A disciplina

Projeto

- Equipe ágil (até 5 pessoas)
- Tema do projeto: Livre
- Entregas:
  - Escopo (Épicos, features, User Stories)
  - Documento de Arquitetura
  - Backlog do produto/sprints
  - Protótipo de baixa/alta fidelidade do projeto
  - Gitpage da Solução

Avaliação

Três categorias de avaliação: "iniciante", "saudável", "maduro" e três aspectos de avaliação: "práticas", "Equipe" e "Projeto"

- **Artefatos avaliados:** (I) Documento de Escopo do projeto (15%), (II) Planejamento/Comunicação Interna e Externa (agenda de trabalho + ferramentas) (10%), Documento de Arquitetura do Projeto (15%), (III) Especificação das histórias de usuários (critérios de aceitação) (20%), (IV) Configuração do repositório de acordo com os padrões de comunidade de software livre (Github) (10%) (V) Protótipo (10%)

- **Práticas avaliadas** (20%): pareamento, produtividade, participação nos rituais, desempenho (commits)

Material de Apoio

- RUBIN, K. S. Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison Wesley, 2012.
- BRECHNER, E. Agile Project Management with Kanban. Microsoft Press, 2015.
- Beck, K., Programacao Extrema (XP) Explicada, 1st ed. Bookman, 2004
- https://engsoftmoderna.info/cap2.html

Hands On

Onboarding

- Checklist Onboarding
  - Contato equipe
  - Canais de comunicação
  - Dias/horários rituais
  - Quadro de conhecimento
  - Configuração git
  - Definição papéis/atribuições
  - Roadmap do projeto (o que estudar, o que fazer)
  - Planejamento da Semana

Checklist Git

- Exemplos/templates de projetos
- Scrum no github - plugin Zenhub
  - Sprint
  - Issues
  - Templates
  - Scrum board

CTEDS

Capacitação Tecnológica em Engenharia e Desenvolvimento de Software

Métodos Ágeis

Introdução -

Professor: Carla Rocha [UnB] *caguiar@unb.br*

01 de Agosto de 2022