

Aula 1

Metodologias Ágeis

Prof. Me. Manoel Flavio Leal

1

Conversa Inicial

2

Manifesto ágil e métodos ágeis

- Tem por objetivo entregar software com celeridade e eficiência
- Destacando a comunicação, colaboração, feedback e a adaptação a mudanças

3

O que aprenderemos

- Conceitos de agilidade
- Cultura organizacional ágil
- Valores e princípios do Manifesto Ágil
- Metodologias integrantes
- *Pair Programming* e refatoração

4

O que é agilidade

5

Definindo agilidade

- De acordo com Jeff Sutherland, "ser ágil é ser capaz de mover-se rápido e facilmente; ser flexível; dinâmico na tomada de decisões; capaz de mudar de direção de forma rápida e eficiente; ser adaptável"



curiosity/Shutterstock

6

Agilidade no ambiente de projetos

- "A agilidade é a habilidade de se adquirir velocidade e flexibilidade no gerenciamento de projetos por meio da adoção de práticas de gestão adequadas ao ambiente e ao tipo de projeto" (Eder et al., 2010)

Equipes ágeis

- Uma abordagem colaborativa
- Orientada a resultados
- Prioriza a entrega de valor ao cliente
- Aprendizagem contínua
- Melhoria iterativa



JUVART/Shutterstock

Onde aplicar

- Desenvolvimento de software
- Gerenciamento de projetos
- Gestão de Produtos
- Marketing
- Recursos Humanos

Vantagens da agilidade em projeto

- Entrega constante de valor
- Flexibilidade
- Maior colaboração
- Melhoria contínua
- Melhor gestão de riscos



Desvantagens da agilidade em projeto

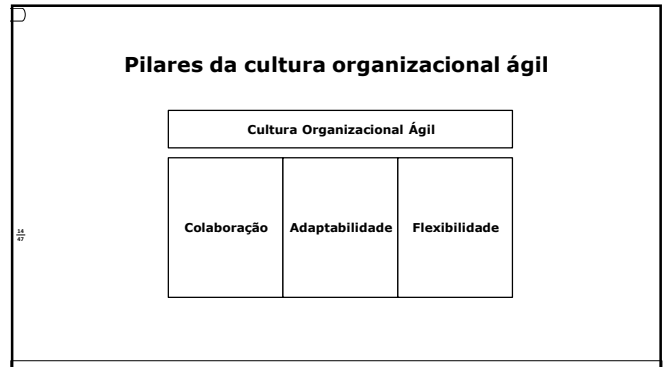
- Menos documentação
- Necessidade de comunicação constante
- Dificuldade de prever prazos e custos



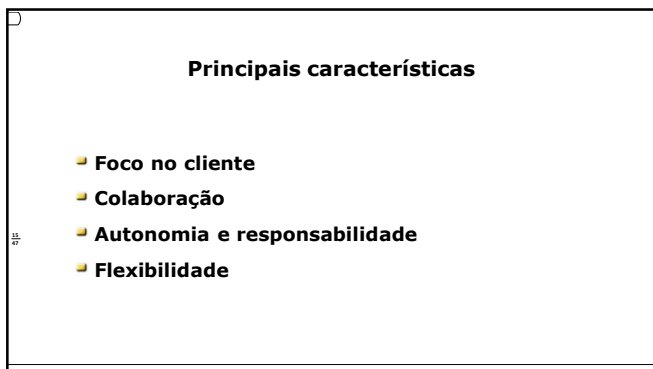
- "A agilidade é um movimento global de negócios que permite que as empresas entreguem produtos e serviços de maneira mais rápida, barata e eficiente. O ágil representa uma mudança de paradigma da administração de projetos, da gestão de produtos e do pensamento em geral" (Sutherland, 2015)



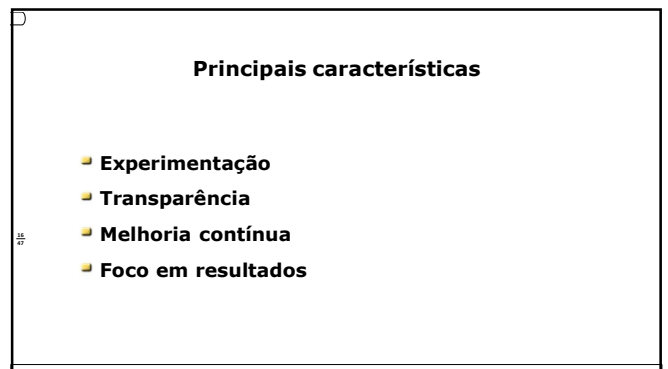
13



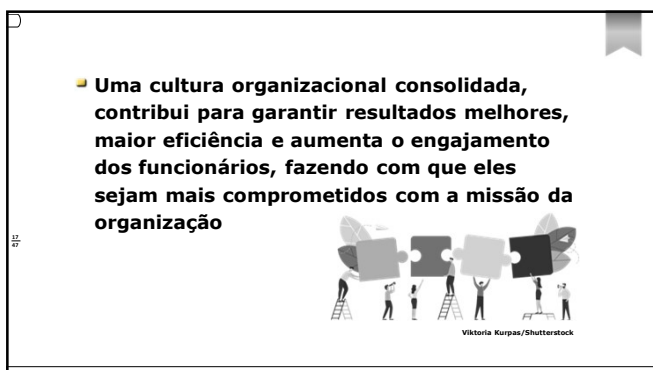
14



15



16



17



18

Conceitos

- Criado em fevereiro de 2001 por um grupo de desenvolvedores em Utah, EUA
- Objetivo de melhorar a forma como software era desenvolvido
- Resultado: Manifesto para Desenvolvimento Ágil de Software

19

Manifesto Ágil

- Criado pelos 17 desenvolvedores da “Aliança Ágil”
- Destaque para a abordagem colaborativa e auto-organizada
- Valores e princípios essenciais para o desenvolvimento de software

20

Onde encontrar

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the left, we value the items on the right more.

Authors:

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekom	Andrew Hunt	Ken Schwaber
Alastair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kohn	
Martin Fowler	Brian Marick	

Postcards from the Postmodern
 Ken Schwaber
 Jeff Sutherland

Portland
 Bangalore
 Kolkata
 Prague
 ...and 200's
 of others
 Shyamkiran
 Shyamkiran
 Shyamkiran

Font: <https://agilemanifesto.org/>

21

Valores

O diagrama apresenta dois blocos de texto, cada um dentro de uma seta. A seta da esquerda aponta para cima e contém os seguintes itens: 'Pessoas e sua interação', 'Software funcional', 'Colaboração com o cliente' e 'Resposta a mudanças'. A seta da direita aponta para baixo e contém: 'Processos e ferramentas', 'Documentos abrangentes', 'Negociação de contratos' e 'Seguir um plano'.

Pessoas e sua interação

Software funcional

Colaboração com o cliente

Resposta a mudanças

Processos e ferramentas

Documentos abrangentes

Negociação de contratos

Seguir um plano

22


Valores

- **VA1 – Indivíduos e interações mais que processos e ferramentas**
- **VA2 – Software em funcionamento mais que documentação abrangente**
- **VA3 – Colaboração com cliente mais que negociação de contratos**
- **VA4 – Responder a mudanças mais que seguir um plano**

23

Princípios

- **PA1 – Satisfazer o cliente através da entrega antecipada e contínua de software de valor**
- **PA2- Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento**
- **PA3 – Fornecer software de trabalho com frequência, com preferência pela menor escala de tempo**



GSP Design/Shutterstock

24

Princípios

- PA4 – Trabalho conjunto entre pessoas relacionadas a negócios e desenvolvedores
- PA5 – Construir projetos ao redor de indivíduos motivados, fornecendo suporte adequado
- PA6 – Priorizar a conversa cara a cara como o método mais eficiente de transmitir informações



25

Princípios

- PA7 – Software funcional como medida primária de progresso
- PA8 – Promover um ambiente sustentável para a equipe e os usuários
- PA9 – Atenção contínua à excelência técnica e bom design

26

Princípios

- PA10 – Maximizar a quantidade de trabalho que não precisou ser feito
- PA11- Arquiteturas, requisitos e designs emergem de times auto-organizáveis
- PA12 – Refletir regularmente sobre como ficar mais efetivo e ajustar o comportamento

27

- O Manifesto Ágil revolucionou o desenvolvimento de software
- Valores e princípios orientam equipes rumo à agilidade
- Benefícios tangíveis para o cliente, equipe e negócio

28

Processos Ágeis

29

Definindo Processos Ágeis

- Técnicas de gerenciamento de projetos e desenvolvimento de software
- Priorizam adaptação, colaboração, entregas interativa e incremental
- Menos burocráticos do que os métodos tradicionais

30

Processos e o Manifesto Ágil

- Base no Manifesto Ágil
- Valores e princípios
- Possuem diversas metodologias

31

Metodologias Ágeis

- Scrum: iterações de trabalho (Sprints)
- Kanban: visualização do fluxo de trabalho
- Lean: minimização de desperdícios
- *Extreme Programming* (XP): qualidade do código e colaboração intensiva
- Crystal: entrega incremental, simplicidade e qualidade do produto

32

Scrum

- Metodologia ágil mais popular
- Sprints de 1 a 4 semanas
- Papéis: *Scrum Master*, *Product Owner*, Time de Desenvolvimento
- Artefatos: *backlog* do produto, *backlog* da sprint



Bakhtiar Zein/Shutterstock

33

Kanban

- Metodologia de gestão com foco na visualização do fluxo de trabalho
- Quadro Kanban e cartões representando tarefas
- Divisão por colunas que representam as etapas do processo



Bakhtiar Zein/Shutterstock

34

Lean

- Baseado na filosofia Lean de gestão de produção
- Minimização de desperdícios e maximização do valor para o cliente
- Identificação e eliminação de retrabalho, espera e excesso de processos



Sabelskaya/Shutterstock

35

Extreme Programming (XP)

- Criada em 1990
- Ênfase na qualidade do código, colaboração e satisfação do cliente
- Práticas comuns: programação em pares, testes automatizados, integração contínua, design simples, reuniões diárias



mentalmind/Shutterstock

36

Crystal

- Criada por Alistair Cockburn
- Indicada para equipes com até 8 pessoas e projetos de pequeno e médio porte
- Valoriza entrega incremental, simplicidade e qualidade do produto



aurielaki/Shutterstock

37

- Processos Ágeis promovem adaptabilidade, colaboração e entregas evolutivas
- Compartilham os valores e princípios do Manifesto Ágil
- Amplamente utilizados em projetos de desenvolvimento de software
- Escolha da metodologia depende das características do projeto e equipe

38

Ferramentas

39

Pair programming

- Colaboração entre dois programadores em um computador
- Piloto e observador/navegador
- Benefícios: qualidade do código, redução de erros, produtividade, compartilhamento de conhecimento, comunicação e colaboração



GoodStudio/Shutterstock

40

Vantagens do pair programming

- Aumento da qualidade do código
- Maior produtividade
- Melhora na comunicação e colaboração
- Maior aprendizado e compartilhamento de conhecimento

41

Desvantagens do pair programming

- Maior custo
- Possibilidade de conflitos e divergências
- Desigualdade na contribuição dos desenvolvedores

42

Refatoração

- Melhoria contínua do código sem alterar seu comportamento externo
- Objetivos: legibilidade, facilidade de manutenção, eliminação de código duplicado, simplificação de estruturas complexas, modularidade e flexibilidade

43

Importância da refatoração

- Prática recorrente no desenvolvimento ágil
- Evita o acúmulo de débito técnico
- Melhora o design do software existente
- Maior facilidade de entendimento, eficiência, menor acoplamento e maior coesão

44

Princípios da refatoração

- Realizar alterações graduais no código
- Assegurar testes para garantir a estabilidade do software
- Reorganização do código, eliminação de duplicação, melhoria da legibilidade, simplificação de algoritmos e melhoria da estrutura de dados

45

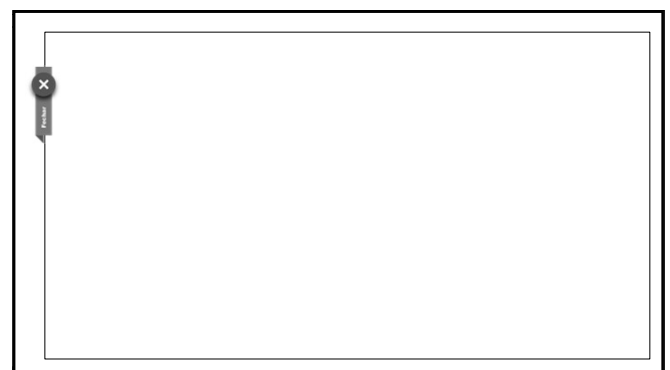
Riscos da refatoração

- Mudanças significativas podem introduzir novos problemas
- Refatorações menores e mais frequentes são menos arriscadas

46

- *Pair programming* e refatoração são práticas essenciais no desenvolvimento ágil de software
- Promovem a colaboração, qualidade do código e melhoria contínua
- Equipes ágeis podem aproveitar os benefícios e superar os desafios dessas ferramentas
- Contribuem para o sucesso de projetos ágeis e entrega de valor incremental

47



48