Métodos Ágeis
Scrum e XP

Prof. Antonio Guardado

Manifesto Ágil - Princípios

- ***Indivíduos e interações*** são mais importantes que *processos e ferramentas.*
- ***Software funcionando*** é mais importante do que *documentação completa e detalhada*.
- ***Colaboração com o cliente*** é mais importante do que *negociação de contratos*.
- **Adaptação a mudanças** é mais importante do que *seguir o plano inicial*.

WebSite: http://www.agilemanifesto.org/

- Métodos ágeis (AM) são uma coleção de metodologias baseadas na prática para modelagem efetiva de sistemas baseados em software. É uma filosofia onde muitas metodologias se encaixam.
- As metodologias ágeis aplicam uma coleção de práticas, guiadas por princípios e valores que podem ser aplicados por profissionais de software no dia a dia.

O que são os Modelos Ágeis?

- Um modelo ágil é um modelo bom o suficiente, nada mais, o que implica que ele exibe as seguintes características:
  1. **Ele** atende seu propósito
  2. **Ele** é inteligível.
  3. **Ele** é suficientemente preciso.
  4. **Ele** é suficientemente consistente.
  5. **Ele** é suficientemente detalhado.
  6. **Ele** provê um valor positivo.
  7. **Ele** é tão simples quanto possível.

O que é (e não é) método ágil?

1. É uma atitude, não um processo prescritivo.
2. É um suplemento aos métodos existentes, ele não é uma metodologia completa.
3. É uma forma efetiva de se trabalhar em conjunto para atingir as necessidades das partes interessadas no projeto.
4. É uma coisa que funciona na prática, não é teoria acadêmica.

O que é (e não é) métodos ágeis? (cont.)

5. É para o desenvolvedor médio, mas não é um substituto de pessoas competentes.
6. Não é um ataque à documentação, pelo contrário aconselha a criação de documentos que tem valor.
7. Não é um ataque às ferramentas CASE.

SCRUM

Processo de Desenvolvimento de Software

- **Scrum** é um processo para construir software incrementalmente em ambientes complexos, onde os **requisitos não são claros ou mudam com muita freqüência.**
- Em Rugby, **Scrum** é um time de oito integrantes que trabalham em conjunto para levar a bola adiante no campo. Ou seja: **times trabalhando como uma unidade altamente integrada com cada membro desempenhando um papel bem definido e o time inteiro focando num único objetivo**.

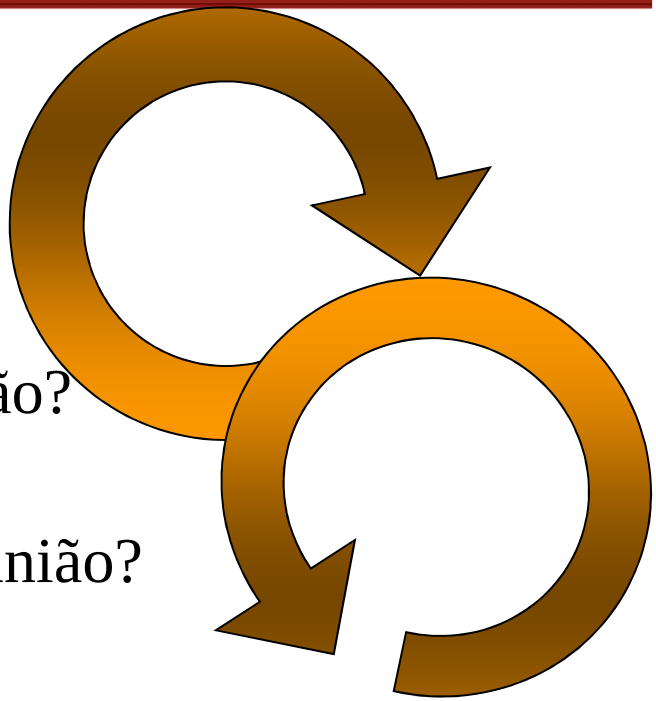
•Formação do Rugby : SCRUM

- O objetivo do **Scrum** é fornecer um processo conveniente para projetos e desenvolvimento orientado a objetos.
- A metodologia é baseada em princípios semelhantes aos de XP: equipes pequenas, requisitos pouco estáveis ou desconhecidos, e iterações curtas para promover visibilidade para o desenvolvimento.

- No entanto, as dimensões em **Scrum** diferem de XP.
- **Scrum** divide o desenvolvimento em Sprints de 30 dias. Equipes pequenas, de até 7 pessoas, são formadas por projetistas, programadores, engenheiros e gerentes de qualidade. Estas equipes trabalham em cima de funcionalidade (os requisitos, em outras palavras) definidas no início de cada Sprint. A equipe toda é responsável pelo desenvolvimento desta funcionalidade

- Todo dia, é feita uma reunião de 15 minutos onde o time expõe à gerência o que será feito no próximo dia, e nestas reuniões os gerentes podem levantar os fatores de impedimento, e o progresso geral do desenvolvimento.

Fases – Sprint
Reuniões Diárias



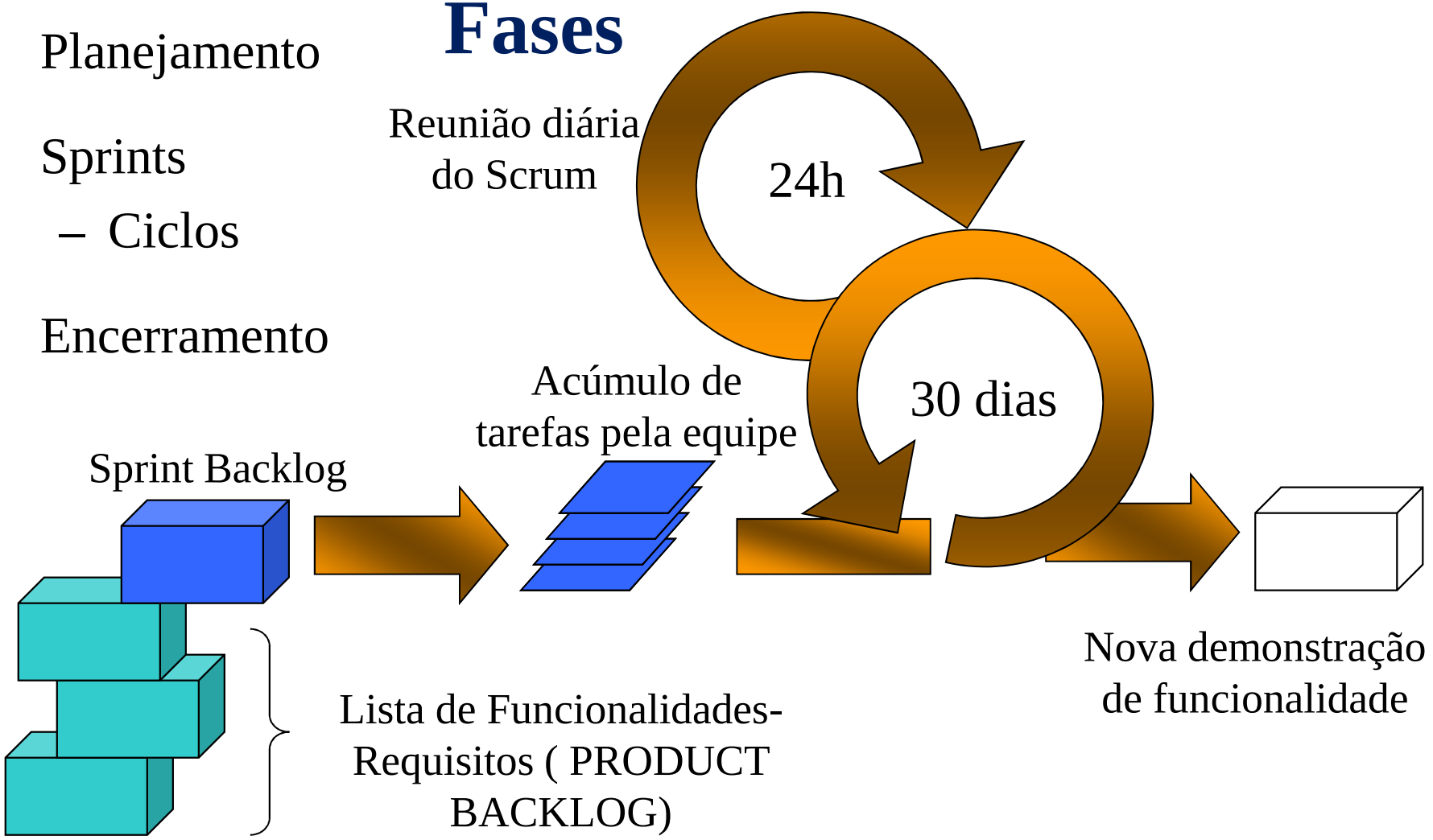
- Todos respondem às perguntas:
  - O que você realizou desde a última reunião?
  - Quais problemas você enfrentou?
  - Em que você trabalhará até a próxima reunião?
- Benefícios:
  - Maior integração entre os membros da equipe
  - Rápida solução de problemas
    - Promovem o compartilhamento de conhecimento
  - Progresso medido continuamente
    - Minimização de riscos

- **Scrum** é interessante porque fornece um mecanismo de informação de status que é atualizado continuamente, e porque utiliza a divisão de tarefas dentro da equipe de forma explicita.

★★★ **Scrum e XP** são complementares pois Scrum provê práticas ágeis de gerenciamento enquanto XP provê práticas integradas de engenharia de software.

Fases

- Planejamento
- Sprints
  - Ciclos
- Encerramento



Fases de encerramento

- Iniciada quando todos os aspectos são satisfatórios (tempo, competitividade, requisitos, qualidade, custo)

- Atividades:
  - Testes de integração
  - Testes de sistema
  - Documentação do usuário
  - Preparação de material de treinamento
  - Preparação de material de marketing

Qualidade, Gerenciamento e Testes

- Passos e papéis bem definidos
- Gerenciamento de riscos
- Revisões freqüentes / diárias
- Definição de padrões
- Realização de testes
- Elaboração de documentação

Controles

Backlog
***Release*/Melhoria**
Mudanças
Problemas
Soluções

•Exemplo de Product Backlog

Product Backlog					
ID	Nome	Importância	Estimativa	Como Demonstrar	Notas
1	Cadastro dos Clientes	40	4	Logar-se, abrir a tela do cliente e escolher entre Novo,Atualizar ou Consultar.Dependendo da opção preencher novos dados ou então digitar a busca pelo nome ou CPF. Se for novo cliente fazer a busca depois para testar se cadastrou.	Fazer o diagrama de sequência com as 3 opções. Não haverá exclusão de dados, o cliente fica INATIVO.
2	Cálculo dos valores do pedido	80	5	Logar-se , abrir a tela de pedidos e buscar pelo número. No grid dos itens incluir um novo item ou atualizar alguma quantidade ou desconto dos já existentes para validar o recálculo dos valores.	Fazer o diagrama de sequência e um diagrama de atividades para o fluxo dos itens. Somente pedidos com a situação EM ANDAMENTO podem ter atualização de itens. Se estiver SEPARADO OU DESPACHADO desabilita as opções de NOVO ou ATUALIZAÇÂO no grid dos itens.
3	etc...				

•Exemplo de Sprint Backlog

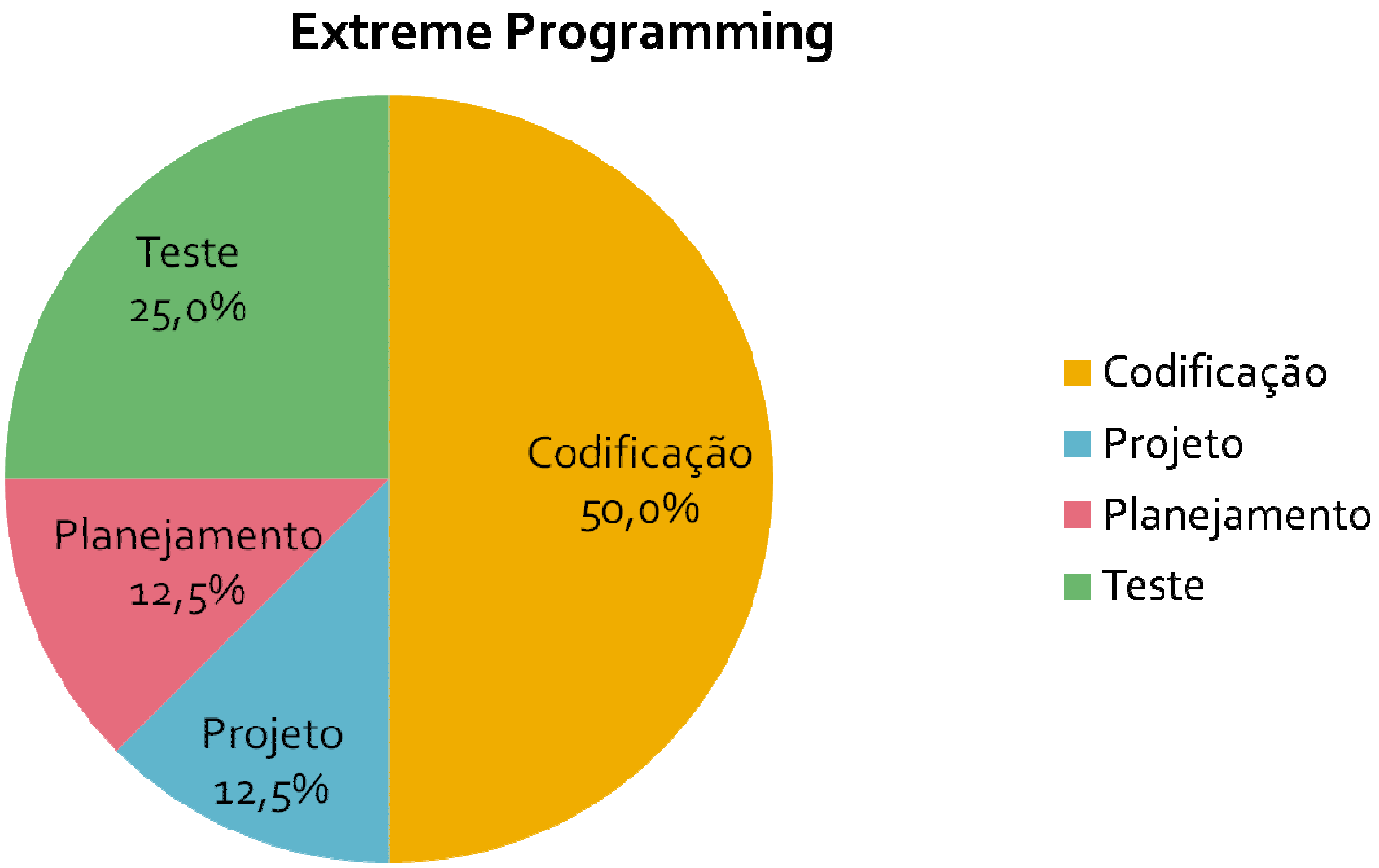
Sprint Backlog										
ID	ID Tarefa	Tarefa	Responsável	Estimativa	Status	Dia 1	Dia 2	Dia 3	Dia 4	etc...
1	1	Montar Diagrama de sequência	José Jr.	1	Completo	100				
	2	Criar método para CRUD	Valquiria	3	Em andamento	30	30			
	3	Definir Interface	Célia	1	Em andamento	0	80			
	4	Verificar estrutura BD	Paulo	1	Completo	100				
	5	Testar e validar	Valquiria	1	Não Iniciado	0	0			

•Sprint Backlog

Extreme Programming - XP

Processo de Desenvolvimento de Software

- A Programação Extrema é uma das metodologias ágeis mais conhecidas. Foi criada por Kent Beck.
- Baseada em cinco **valores,** alguns **princípios** e várias **práticas** que ocorrem no contexto de quatro atividades. Ela se destina a times de até dez programadores, projetos de curto e médio prazo.


Extreme Programming
Teste
25,0%
Codificação
50,0%
Planejamento
12,5%
Projeto
12,5%
Codificação
Projeto
Planejamento
Teste

XP (Extreme Programming) - Valores

- **Comunicação** – para um projeto de sucesso é necessária muita interação entre os membros da equipe, programadores, cliente, treinador. Para desenvolver um produto, o time precisa ter muita qualidade nos canais de comunicação. Conversas presenciais são sempre melhores do que telefonemas, e-mails, cartas ou fax.
- **Feedback** – as respostas às decisões tomadas devem ser rápidas e visíveis. Todos devem ter, o tempo todo, **consciência** do que está acontecendo.
- **Coragem** – alterar um código em produção, sem causar bugs, com agilidade, exige muita coragem e responsabilidade.

XP (Extreme Programming) - Valores

- **Simplicidade** – para atender rapidamente às necessidades do cliente, quase sempre um dos valores mais importantes é simplicidade. Normalmente o que o cliente quer é muito mais simples do que aquilo que os programadores constroem.
- **Respeito** – todos têm sua importância dentro da equipe e devem ser respeitados e valorizados. Isso mantém o trabalho energizado.

XP (Extreme Programming) - Papéis

- **Programadores** - foco central da metodologia, sem hierarquia.
- **Treinador** (ou *coach*) - pessoa com mais experiência no time, responsável por lembrar os outros das regras do jogo (que são as práticas e os valores de XP). O treinador não precisa necessariamente ser o melhor programador da equipe e sim o que mais entende da metodologia XP.

XP (Extreme Programming) - Papéis

- **Acompanhador** (ou *tracker*) - responsável por trazer para o time dados, gráficos, informações que mostrem o andamento do projeto e ajudem a equipe a tomar decisões de implementação, arquitetura e design. Algumas vezes o próprio coach faz papel de tracker. Outras o time escolhe sozinho quem exercerá este papel.
- **Cliente** – em XP o cliente faz parte da equipe. Deve estar sempre presente e pronto para responder às dúvidas dos programadores.

XP (Extreme Programming)

Existe um grande ênfase ao **trabalho em duplas**, aonde um analista mais experiente trabalha com um novato. Enquanto o mais jovem trabalha na programação o mais antigo vai revisando o código. Dessa forma ao mesmo tempo desenvolve-se a equipe, e melhora-se automaticamente a qualidade do código fonte gerado.

•Programação em duplas

XP (Extreme Programming) - Práticas

- **Estórias de uso**
  - Usadas como requisitos do sistema
  - Mesmo propósito dos casos de uso (de UML), porém menores e mais simples
  - Escritos na linguagem do cliente com o mínimo de detalhes para a estimativa de custos
- **Iterações**
  - Desenvolvimento dividido em iterações
  - Possuem duração entre 1 e 3 semanas
  - Funcionalidades são entregues no final de cada iteração
  - Prazos devem ser levados a sério, negocie o escopo se for necessário

XP (Extreme Programming) - Práticas

- **Jogo do planejamento**
  - Planejamento das iterações:
    - Feito no início de cada iteração
      - Estórias de uso são escolhidas de acordo com a importância pro cliente e o risco pro projeto
      - Estórias são divididas em tarefas de 1 a 3 dias
      - Tarefas são distribuídas entre programadores e estimadas pelos próprios executores
        - Estimativa preferencialmente baseada no passado
        - Leva-se em conta a programação em pares
    - Estórias são adicionadas ou removidas para completar o tempo da iteração

XP (Extreme Programming) - Práticas

- **Versões freqüentes e pequenas**
  - Funcionalidades implementadas são rapidamente entregues ao cliente
  - Permite um feedback mais rápido do cliente reduzindo o impacto de mudanças de requisitos
  - Versão pode ter inclusive uma única iteração
- **Reuniões rápidas (*stand-up meeting*)**
  - Faz a comunicação entre toda a equipe para comunicar problemas, soluções, etc.
  - Reunião feita em pé com poucos minutos de fala para cada integrante

XP (Extreme Programming) - Práticas

- **Testes** - todo desenvolvimento inclui testes. Kent Beck diz que código sem teste não existe. Os testes devem ser escritos de preferência antes do desenvolvimento (TDD - test driven development) e sempre devem rodar de forma automatizada.
- **Refatoração** - é um conjunto de técnicas para modificar o código do sistema sem alterar nenhuma funcionalidade. O objetivo é simplificar, melhorar o design, limpar, enfim, deixar o código mais fácil de entender e dar manutenção.

XP (Extreme Programming) - Práticas

- **Programação Pareada** - em XP dois programadores sentam juntos no mesmo computador e programam juntos. Enquanto um programador digita, o outro observa, pensa em melhorias, alternativas.
- **Propriedade Coletiva** - O código fonte não pertence a um único programador. Todos da equipe são responsáveis. Todos alteram código de todos (mas sempre rodando os testes para se certificar que nada foi quebrado)

XP (Extreme Programming) - Práticas

- **Integração Contínua** - depois de testada, cada nova funcionalidade deve ser imediatamente sincronizada entre todos os desenvolvedores. Quanto mais freqüente for essa integração, menores são as chances de conflitos de arquivos que vários programadores alteram simultaneamente.
- **Semana de 40 horas** - programar é uma atividade intensa e que não rende se o programador não estiver descansado e disposto. Por isso, 40 horas de trabalho por semana é essencial para a saúde do time.

XP (Extreme Programming) - Práticas

- **Cliente Sempre Presente** - o cliente não é alguém de fora, mas sim um membro da equipe. Ele deve estar sempre disponível e pronto para atender às dúvidas dos desenvolvedores.
- **Padronizações** - se todo o time seguir padrões pré-acordados de codificação, mais fácil será manter e entender o que já está feito. O uso de padrões é uma das formas de reforçar o valor **comunicação**.

Abordagem clássica x Abordagem Ágil

	Clássica	Ágil
Desenvolvedor	hábil	ágil
Cliente	pouco envolvido	comprometido
Requisitos	conhecidos, estáveis	emergentes, mutáveis
Retrabalho	caro	barato
Planejamento	direciona resultados	resultados o direcionam
Foco	grandes projetos	projetos de natureza exploratória e inovadores
Objetivo	controlar, em busca de alcançar o planejado	simplificar processo de desenvolvimento