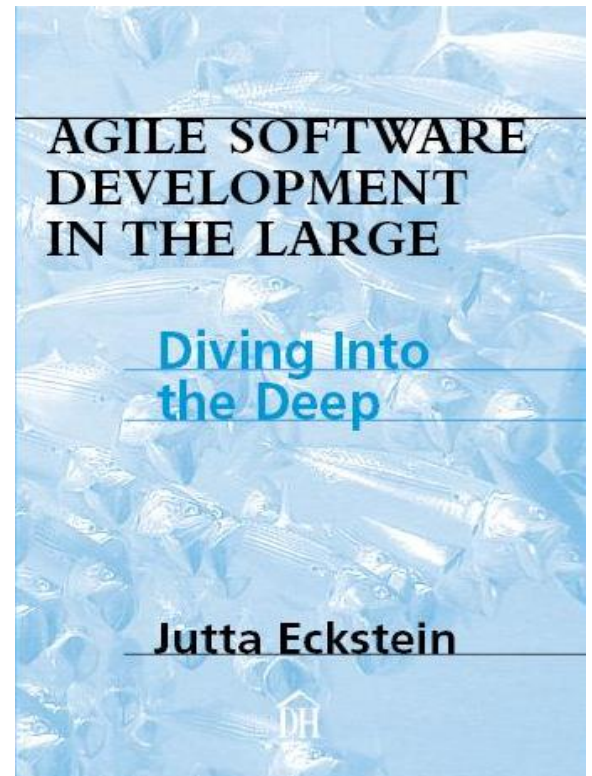


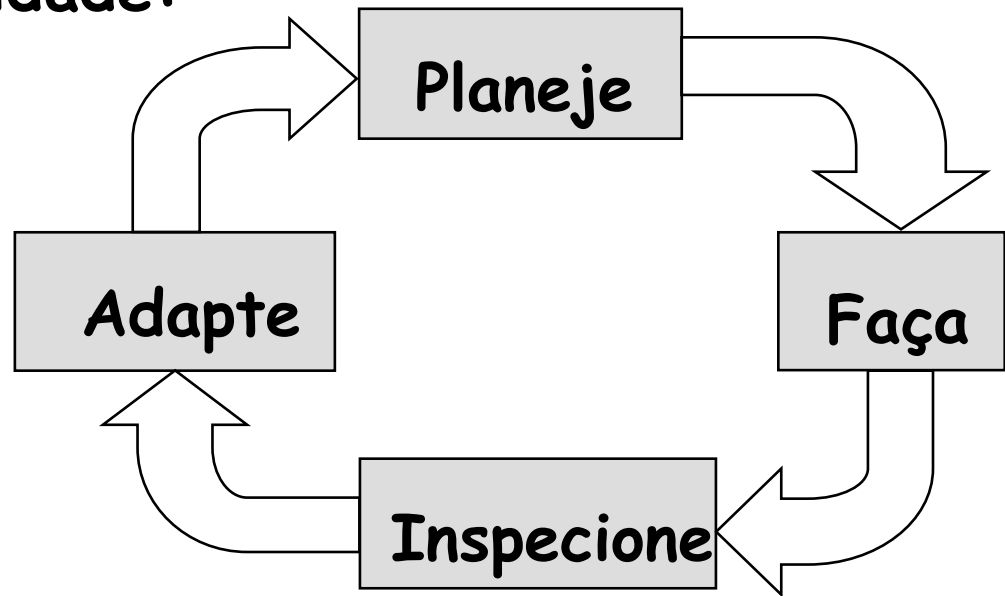
Planejar, Estimar e Corrigir num Projeto Ágil

Jutta Eckstein
Encontro Ágil 2009



Desenvolvimento Ágil

- Planejar, estimar, corrigir e portanto - aprender são atividades em progresso
- Essência da agilidade:



Agrupando Funcionalidades

Funcionalidades

- ou (também conhecido como)

- Histórias de Usuários
- Casos de Uso
- Requisitos

- **Afirmações curtas**

- De funcionalidade
- Do ponto de vista do usuário

⇒ **Uma funcionalidade cria valor de negócio, que é mensurável**

Definindo Funcionalidades

■ Pode-se usar esse modelo

Como um <tipo de usuário> ,

Eu quero <objetivo>

Para que <motivo>

■ Tudo bem usar um formato diferente,

- Mas garanta que todas as funcionalidades
 - Provejam valor de negócio para alguém
 - São mensuráveis
- Portanto cada funcionalidade precisa esclarecer
 - Quem pode aceitá-la
 - Por qual critério

Backlog do Produto

■ Backlog do produto

- "Consiste de todo trabalho que pode ser previsto para um produto." (Scrum)
 - Trabalho é priorizado e estimado
 - Mais detalhes quando chegar a hora
- Qualquer um pode contribuir
- Atualizado e bem exibido
- Uma lista para múltiplos times
- Tem um product owner

Product Owner

■ Responsável por

- Garantir uma visão comum para o projeto
- Estabelecendo prioridades
- Definindo condições de satisfação para a iteração

■ Tipicamente

- Marketing / Gerência de produto
- Usuário / analista
- Financiador do projeto

⇒ **Representa o cliente**

Estimando Funcionalidades

Estimando Funcionalidades

■ Estime complexidade

- E não duração

■ Estimativas relativas

- A funcionalidade A é duas vezes mais difícil que a funcionalidade B
- A funcionalidade C é igualmente difícil que a funcionalidade A
- ...

■ Unidades de Estimativa não importam

- Tempo ideal
- Pontos de funcionalidade (pode ser mais fácil)

Pôquer do Planejamento

- **Estime funcionalidades colaborativamente**
 - Cada um escolhe uma carta de seu baralho ao mesmo tempo
 - Estimativas possíveis:
 - 1, 2, 3, 5, 8
- **Esclareça as funcionalidades para atingir um consenso**
- **Divida funcionalidades se necessário**
- **Repita o processo até atingir um acordo**

Planejamento de Iteração

Iterações (ou Sprints)

- **Duração ou: Quão frequentemente você consegue entregar?**
 - Elas são curtas o suficiente...
 - para lidar com a incerteza das necessidades dos usuários?
 - para manter as prioridades inalteradas?
 - Elas são grandes o suficiente...
 - para realizar alguma coisa mensurável?

- **Examinação**
 - Funcionalidade
 - Estimativas

Como isso pode ser planejado?

■ Expectativa:

- Planeje - desenvolva - entregue

■ Dificuldades:

- Planejamento orientada a atividades
 - Não é realmente mensurável
 - Nem é executável nem testável
- Planejamento orientado a componente
 - Baseado nas dependências dos componentes
 - Difícil de aceitar com testes

■ Portanto:

- Planejamento orientado a resultados

Planejamento orientado a resultado

- **Planeje para cumprir uma funcionalidade de negócio**
 - Cumprir significa entregar
- **Planejamento detalhado apenas para os próximos passos**
 - Estimar e planejar são atividades contínuas e constantes
- **Desenvolvimento sustentável**
 - Quantidade de trabalho deve bater com quantidade de tempo

Planejamento de Iteração (ou backlog do Sprint)

- **Iterações são direcionadas por funcionalidades**
- **Planejamento de iteração de fina granularidade**
 - Responsabilidade dos desenvolvedores
 - Planejar para completar as funcionalidades
 - Estimar funcionalidades quebrando-as em tarefas
 - Estimar apenas tarefas que agregam valor ao projeto
 - Estimar tarefas „naturais“ enquanto elas não forem naturais
 - Responsabilidade do product owner
 - Definir o critério de aceitação

Estimando Iterações

- **1ª iteração: velocidade do projeto é desconhecida**
 - Quanto tempo disponível temos?
 - Exemplo:
 - Tamanho da iteração: 1 semana, 3 desenvolvedores = 15 dias ideais (tempo ideal)
 - Reduzindo de 2,5 = $15 \text{ dias} / 2,5 = 6 \text{ dias reais}$ (tempo real)
 - **O time não aceita mais tarefas para a próxima iteração do que cabem no tempo real**
- **Estimando as próximas iterações**
 - O time terá a mesma velocidade na próxima iteração do que teve na anterior

Dividindo Funcionalidades

- **Planeje para entregar uma funcionalidade de valor**
 - A definição de pronto deve ser estabelecida
- **Algumas vezes a iteração parece ser muito curta para entregar uma funcionalidade**
 - Não separe funcionalidades em tarefas (passos de desenvolvimento)
 - Divida de acordo com o critério de aceitação
- **Tamanho máximo de uma funcionalidade**
 - 2-3 pessoas podem completá-la durante a iteração
 - **Idealmente: um time completa 5-10 funcionalidades por iteração**

Ferramentas de Planejamento

■ Ferramentas conservadoras

- Flip charts
- Cartões
- Planilha

■ Ferramentas sofisticadas

- Trac (<http://trac.edgewall.org>)
- AgileTrac (<http://www.agile-trac.org>)
- PPTS (<http://ses-ppts.sourceforge.net/>)

Acompanhamento de Iteração

- Planos devem ser exibidos em lugares proeminentes
 - De fácil acesso
 - Faça com que o progresso seja visível
 - Legível e alterável por todos
- Acompanhados na sincronização diária

Planejamento de Versão

- **Atinja a versão de produção através de várias versões internas**
- **Versões internas são definidas em termos de funcionalidades ou pacotes de funcionalidades**
- **Planejamento da versão com baixa granularidade**
 - Direcionado pelo product owner
 - Marketing
- **Exibe a visão geral do projeto**

Correções de Curso

Reflexão do Time

Explicitate o progresso

- Mostre software funcionando frequentemente
- Mostre estimativas do time e suas conquistas

Retrospectivas

- Cobrem dois níveis:
 - Nível de desenvolvimento: evolução do software
 - Meta-nível: inspeção do processo

Correções de Curso

■ Software funcionando e retrospectivas

- Melhoria contínua do planejamento

■ Desenvolvimento

- Software funcionando provê feedback direto e concreto

- Integração, qualidade, testes, aceitação do cliente, arquitetura
- Pequenas correções de curso continuamente

■ Processo

- Retrospectivas regulares

- Reflexões permanentes e otimizações do processo
- Melhoria de qualidade nas estimativas e nos planos

Lidando com Mudanças

Magnitudes de Planejamento

■ Quatro variáveis:

- Tempo
- Escopo
- Recurso
- Qualidade

■ Se o escopo pode mudar:

- Pelo menos mais uma variável deve lidar com isso

Agilidade significa Acolher Mudanças

- **Do ponto de vista de um time**
 - Cada iteração é dirigida por funcionalidades de alta prioridade
 - Não há diferença se essas funcionalidades são
 - Velhas, novas, ou terem mudado
- **Desenvolvimento sustentável**
 - A quantidade de trabalho deve bater com a quantidade de tempo
- **Lembre-se:**
 - Sem mudanças durante a iteração

Ajustes

Dirigido por plano versus dirigido por planejamento

- Não existe um plano final no começo do projeto
- Feedback tem um impacto direto no plano

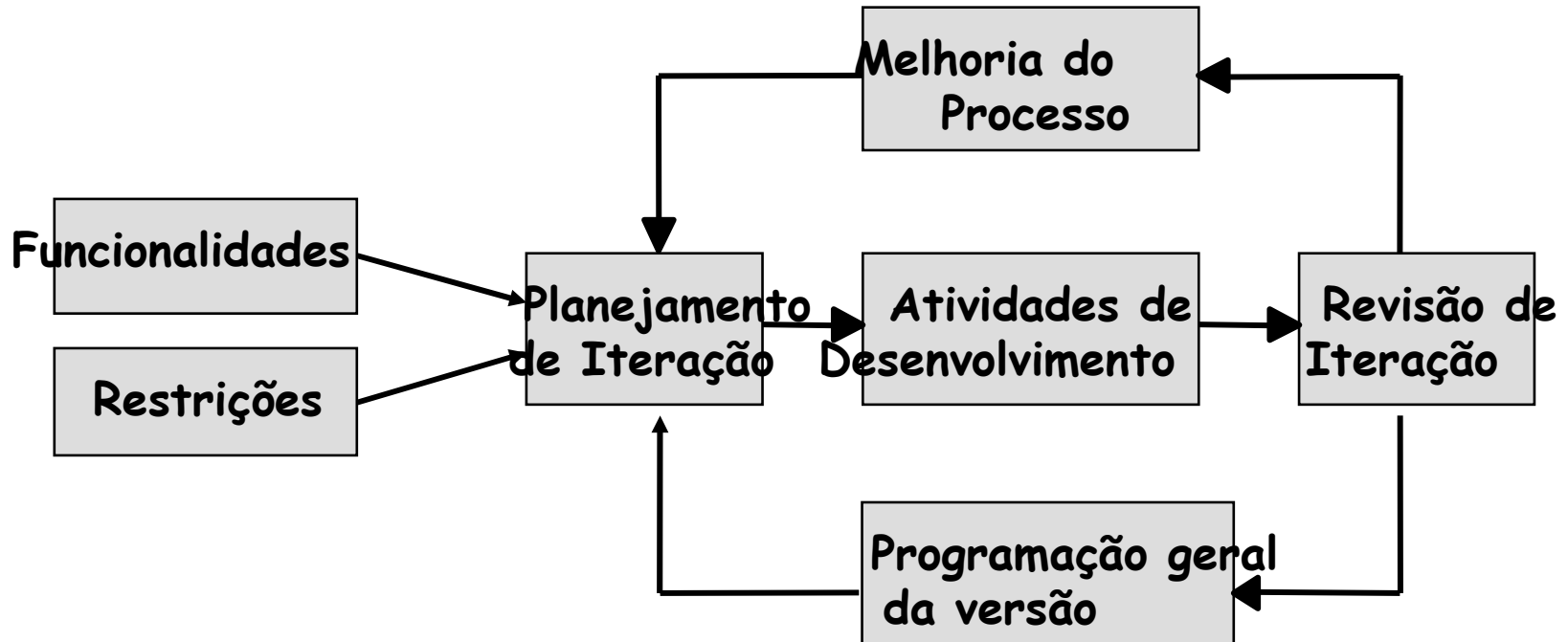
„Um plano não é nada;
planejar é tudo.“

Dwight D. Eisenhower

Resumo

Resumo

- Uma iteração é um ciclo de desenvolvimento de tempo fixo
- Funcionalidades são realizadas durante a iteração



Começando uma iteração

- **O Product Owner apresenta as funcionalidades para a próxima iteração**
- **O time**
 - Estima a complexidade das funcionalidades
 - Divide as funcionalidades em tarefas
 - Estima as tarefas em horas
 - Bate as tarefas estimadas com a disponibilidade e velocidade
- **Comprometimento**
 - O time se compromete com as funcionalidades

Durante uma Iteração

■ O time

- Sincroniza através do papo em pé
- Estima o esforço a ser realizado
- Realiza atividades de desenvolvimento para a entrega

■ Coach / Scrum Master

- Garante o processo
- Garante que os impedimentos são resolvidos

■ Product Owner

- Provê feedback sobre as funcionalidades
- Seleciona novas funcionalidades para a próxima iteração
- (Re-) prioriza o backlog se necessário
- Clarifica o critério de aceitação

Reflexão - Acabando uma Iteração

■ Revisão da Iteração

O product owner confere os objetivos da iteração passada

- O time apresenta o que foi entregue
- O product owner aceita / rejeita o que foi feito
- Todos conferem a velocidade

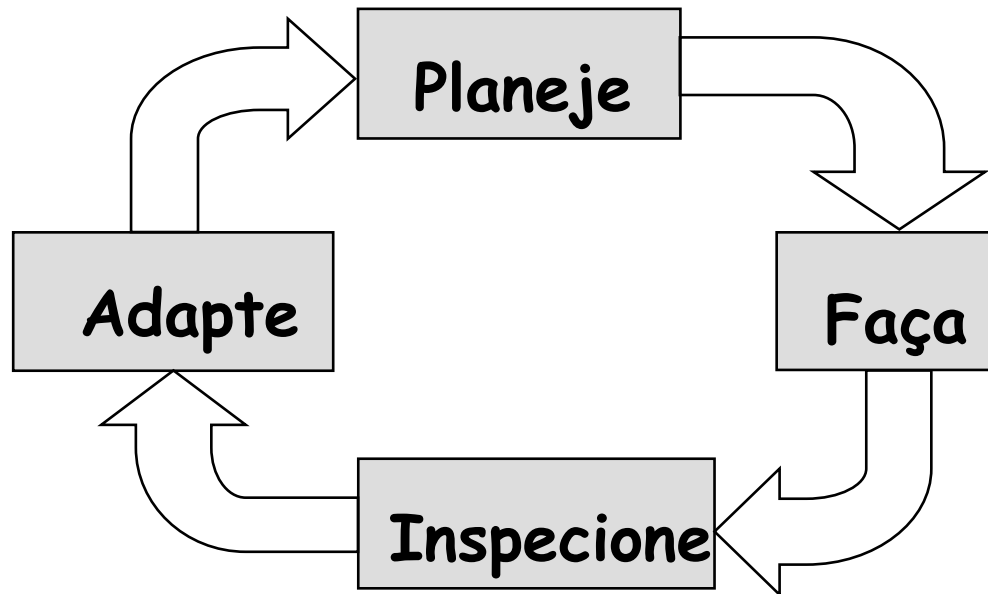
■ Retrospectiva

- O time reflite sobre (e define um plano de ação para os 3 principais) para:
 - Coisas que deram certo
 - Coisas que precisam melhorar
 - Coisas que ainda são difíceis

Cuidado:

„Se um projeto está no prazo e dentro do orçamento isso não significa que ele foi um projeto de sucesso, mas um projeto bem estimado.“

Martin Fowler



Muito Obrigado!

Um novo livro chegando sobre:

**Agile Software with
Distributed Teams
(Métodos Ágeis para
Times Distribuídos)**

Contatos:

Jutta Eckstein

je@it-communication.com

www.it-communication.com

