

# CMP1553 – Melhoria de Processo de Software

Profs. Alexandre Cláudio de Almeida

SCRUM



# Scrum

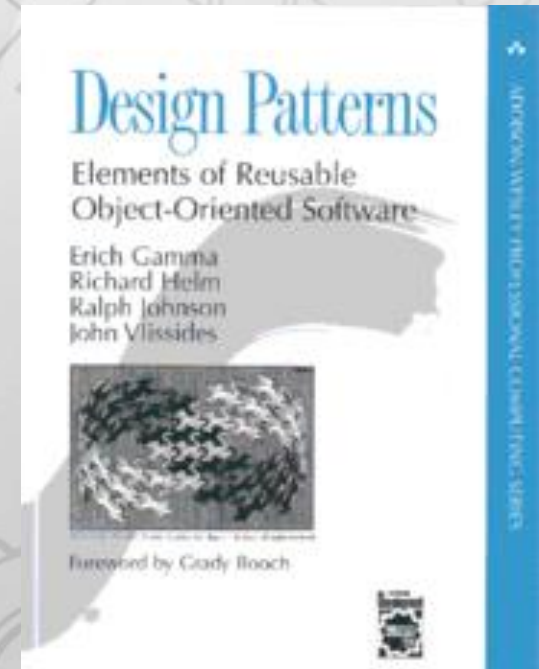
Definição informal:  
Estratégia em um jogo de rugby onde jogadores colocam uma bola quase perdida novamente em jogo através de trabalho em equipe.

# Origens de Scrum

- Jeff Sutherland - [jeffsutherland.com](http://jeffsutherland.com)
- Ken Schwaber - [www.controlchaos.com](http://www.controlchaos.com)
- Mike Beedle - [www.mikebeedle.com](http://www.mikebeedle.com)
- Conferências
  - OOPSLA 96, PLoP 98
- Inspiração
  - Desenvolvimento Iterativo e Incremental em empresas (DuPont) nos anos 80

# Origens de Scrum

- Desenvolvimento de software a partir de padrões de projeto (*design patterns*)



- Mas, o que é isto ???

# O quê são padrões ?

- No final dos anos 70, o arquiteto Christopher Alexander escreveu dois livros com a idéia.
- Cada padrão descreve um problema recorrente no nosso ambiente e, em seguida, o princípio de sua solução.
- A solução pode ser aplicada diversas vezes, nunca da mesma maneira.
- Um exemplo: escritório com janela.

# Fundamentos de Scrum

- Desenvolvimento de software depende muito de criatividade e de trabalho
- Logo, não é um bom candidato a processos pré-definidos
  - modelo de controle de processo empírico
- O desenvolvimento nem sempre será repetitivo e bem definido
- Mas existem padrões que podem ser usados

# História (processos)

- Processo definido
  - Funciona em ambientes controlados
- Processo empírico
  - Processos de controle industrial
    - Babatunde "Tunde" Ogannaiké
      - Process Dynamics, Modeling and Control, 94
  - Funciona para processos
    - Complexos e Imprevisíveis
  - Princípios:
    - **Visibilidade, Inspeção e Adaptabilidade**

# Ênfases

- Comunicação
- Trabalho em equipe
- Flexibilidade
- Fornecer software funcionando
  - incrementalmente



# Principais Padrões

- *Backlog*
- Equipes
- *Sprints*
- Encontros Scrum
- Revisões Scrum/Demos



# *Backlog*

- Lista de todas as funcionalidades desejadas
- É gerada incrementalmente
  - Começa pelo básico, o extra aparece com o tempo
- Pode conter
  - Tarefas diretas, casos de uso e histórias (a la XP)
- A lista é priorizada pelo dono do projeto
  - Cliente, depto de marketing, ...

# O *Backlog* Inicial

- Deve conter características que agreguem algum valor de negócio ao produto
- Novos requisitos aparecem quando o cliente vê o produto
- A arquitetura do sistema surge enquanto o projeto surge e é refatorado

# Equipes

- Sem nível hierárquico nem papéis
  - Mas com várias especialidades
- Estão todos no mesmo barco
- Geralmente equipes pequenas (até 10)
  - Existem casos com equipes maiores (800 !)
  - Usa-se também Scrum hierárquico
- Comunicação é essencial
  - Encontro Scrum diário

# *Sprint*

- Unidades básicas de tempo (até 30 dias)
- Começa com um encontro *Sprint*
  - Tarefas do *Backlog* são priorizadas
  - A equipe seleciona tarefas que podem ser completadas durante o próximo *Sprint*
  - As mesmas podem ser quebradas para o *Backlog* do *Sprint*
  - Cada tarefa recebe um responsável na equipe
  - Não há mudança nas tarefas durante o *Sprint*

# Encontro Scrum 1/2

- Pequenos encontros diários da equipe
  - geralmente pela manhã
  - galinhas e porcos (só os porcos falam)
  - todos os porcos devem participar
- Questões que aparecem devem ser resolvidas durante o dia e não na reunião
- Os encontros iniciais são geralmente mais longos

# Encontro Scrum 2/2

- Questões que devem ser respondidas por cada porco:
  - 1) O quê você fez ontem?
  - 2) O quê você vai fazer hoje?
  - 3) Quais os problemas encontrados?
- Ajuda a manter as promessas
- Evita: Como um projeto atrasa um ano?
  - Um dia por vez ...
  - Qualquer deslize pode ser corrigido de imediato

# Local do Encontro

- Sempre o mesmo local e hora
- Pode ser o local de desenvolvimento
- Pessoas sentadas ao redor de uma mesa
- A sala já deve estar arrumada antes
- Punições (atrasos/faltas)
- Todos devem participar
- Galinhas ficam na periferia
- Pode ser em pé
- Sala bem equipada, quadro branco, etc.



# Revisão do *Sprint*

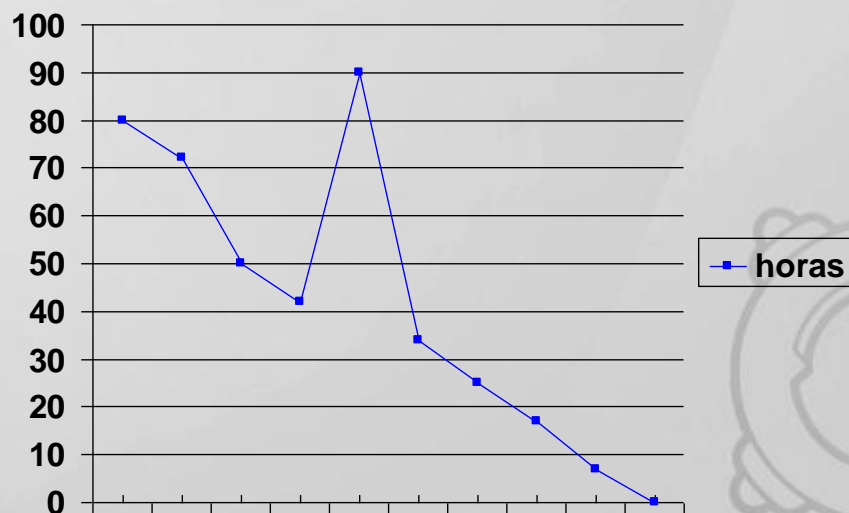
- No final de cada *Sprint* é feita uma reunião com todos os interessados
- Geralmente
  - Na forma de demonstração
  - Informal (preparação rápida, sem projetor,..)
  - Deve ser o resultado natural de um *Sprint*
- O projeto é comparado com os objetivos iniciais do *Sprint*

# Scrum Master 1/2

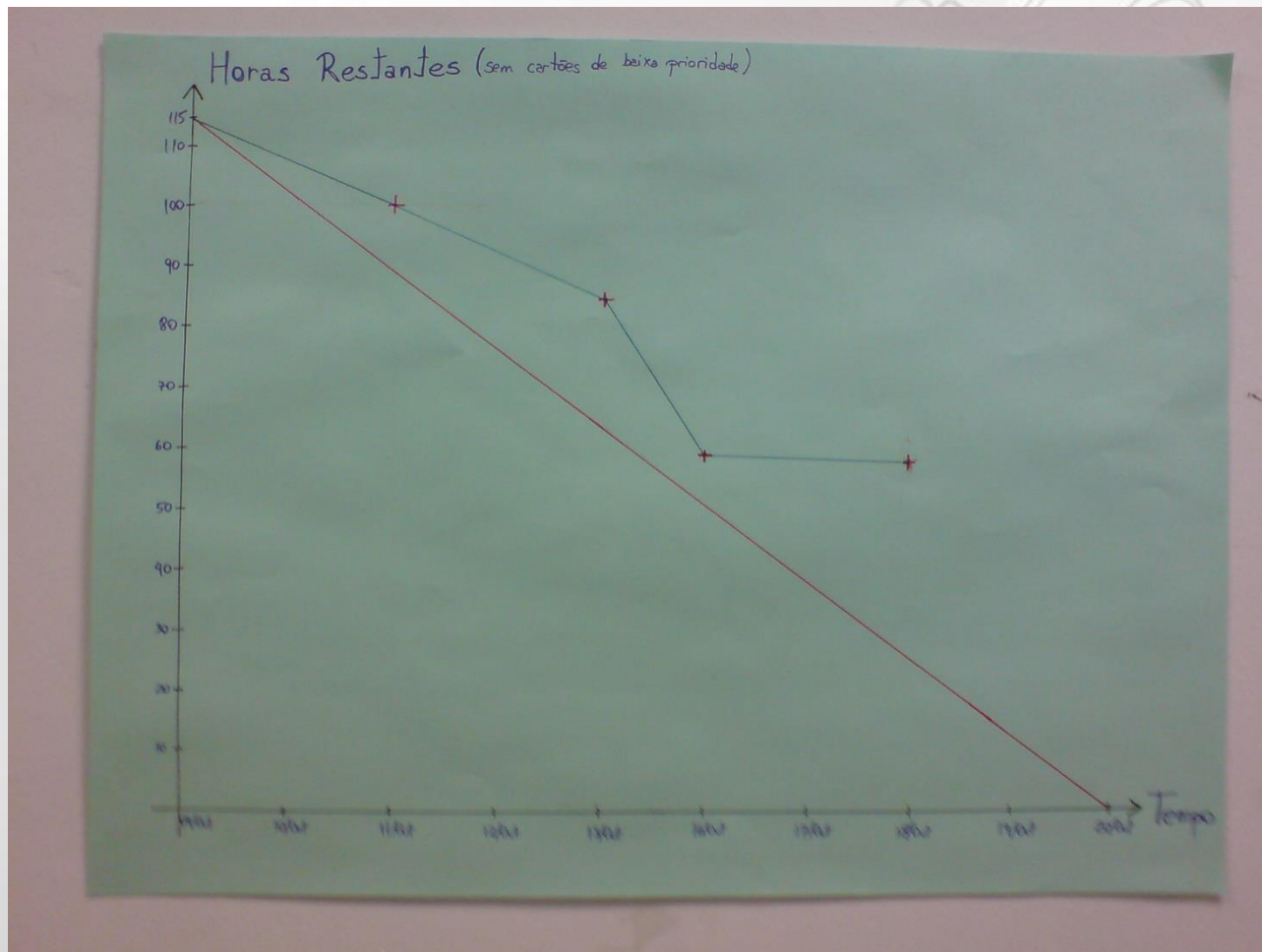
- Faz com que a equipe viva os valores e práticas de Scrum
- Protege a equipe de:
  - Riscos e interferências externos
  - Excesso de otimismo
- Resolve os problemas que aparecerem
  - logísticos
  - de conhecimento/habilidade

# Scrum Master 2/2

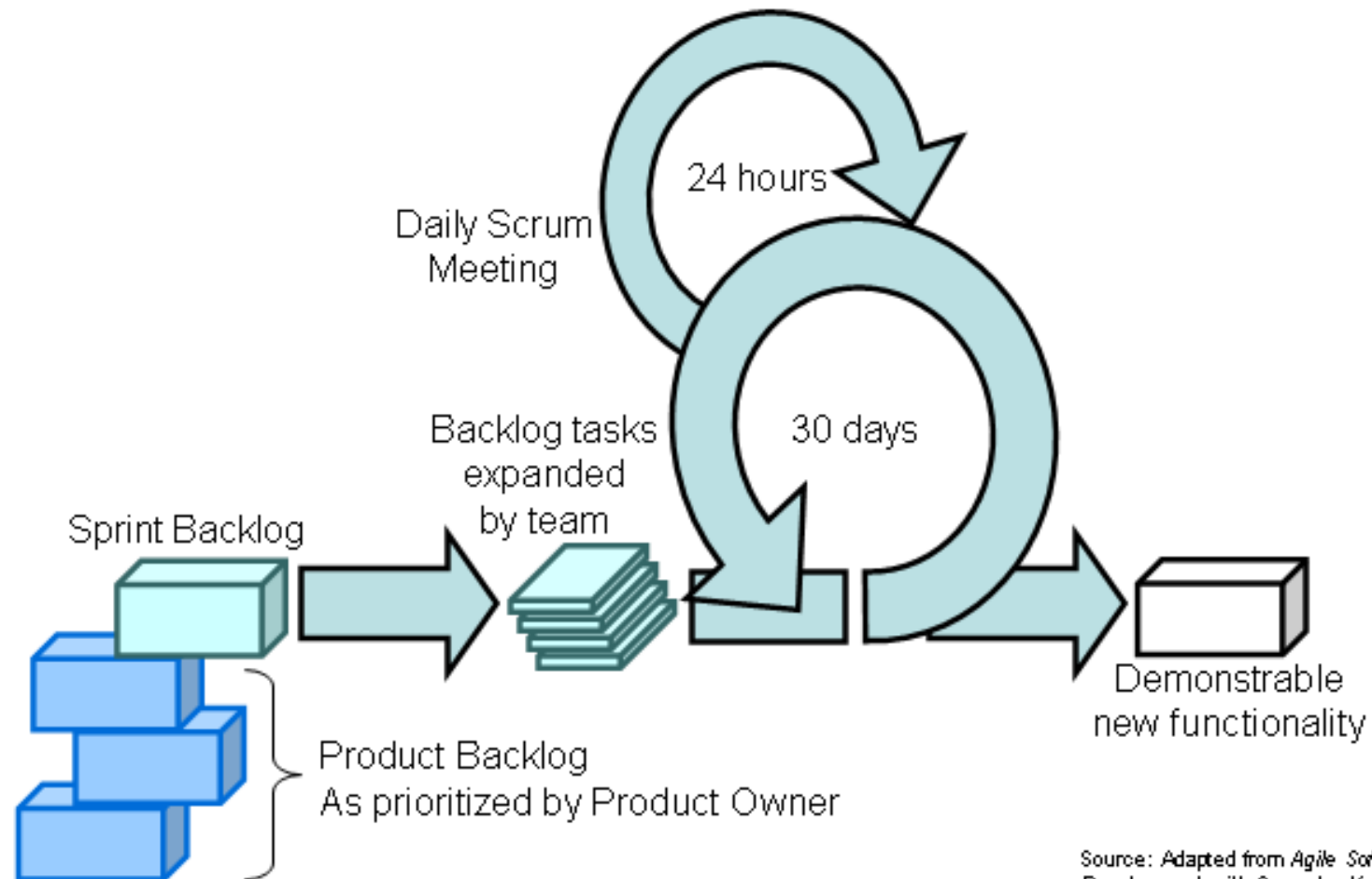
- Mantém o *Backlog* do *Sprint*
  - Tarefas completadas
  - Identifica eventuais problemas
- Mantém um gráfico de “quanto falta”



# Exemplo real



# Scrum de Forma Gráfica



Source: Adapted from *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.

# Scrum Final

- Não faz parte do Scrum padrão, mas é bem usado na prática
- Um último *Sprint* para “fechar” o produto
- O objetivo é:
  - Preparar a versão de produção
  - O foco é a eliminação de erros

# Nem tudo são flores

Scrumm are the most dangerous phase in rugby, since a collapse or improper engage can lead to a front row player damaging or even breaking his neck.  
(Wikipedia)

# Scrum Distribuído

- Para grandes sistemas e grandes equipes
- É atualmente um tema em discussão (Jeff Sutherland – Agile2006)
- Possui novas dificuldades



# Principais Dificuldades

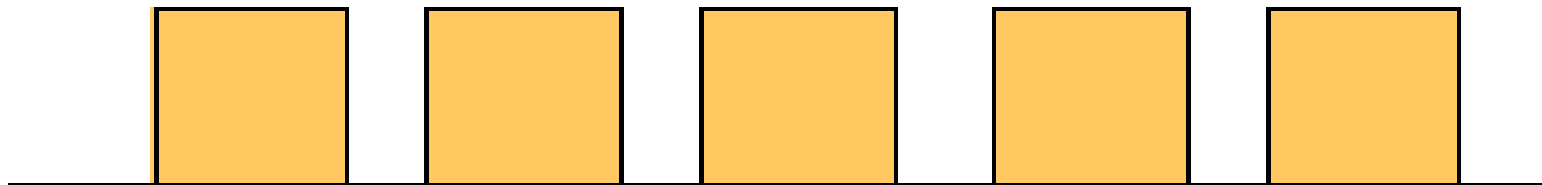
- Independência de equipes
- Problemas de comunicação
- Barreiras Culturais
- Modo de Trabalho
- Práticas de Scrum são para equipes reunidas

# Práticas Distribuídas

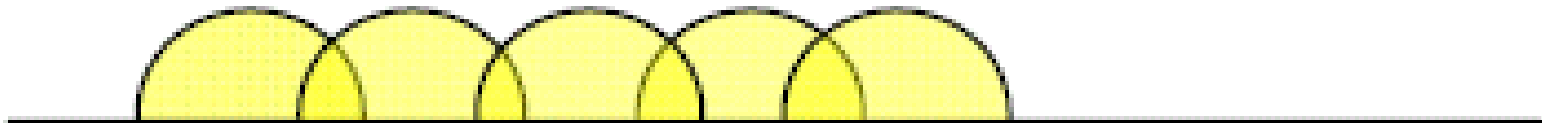
- Particionar o trabalho eliminando dependências
- Fortalecer a comunicação
- Estabelecer Padrões



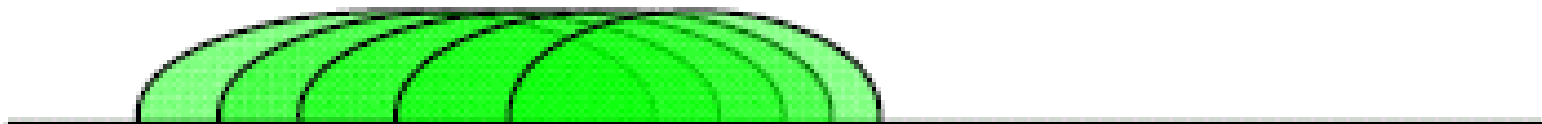
# Tipos de Scrum Distribuído



**Type A – Isolated Scrums**



**Type B – Scrum of Scrums Overlaps Scrums Teams**



**Type C – Totally Integrated Scrums**

# Caso de Scrum Distribuído

- SirsiDynix & StarSoft
- Objetivo: Produzir um sistema integrado e complexo para bibliotecas
- Mais de 50 desenvolvedores
- Mais de 1.000.000 de linhas de código
- Alocados em: EUA, Canadá e Rússia

# Práticas Adicionais

- Reunião diária com todos do desenvolvedores
- Reunião diária com o dono do produto
- Builds automáticos a cada hora
- Pradronização dos desenvolvedores
- Inserção de práticas de XP

# Mais Informações

- Agile Alliance - [www.agilealliance.org](http://www.agilealliance.org)
  - Ótima fonte sobre métodos ágeis
- Scrum Alliance - [www.scrumalliance.org/](http://www.scrumalliance.org/)
- Mountain Goat Software
  - [www.mountangoatsoftware.com](http://www.mountangoatsoftware.com)
  - Site de um treinador de *Scrum Masters*
- Site do Ken Schwaber - [www.controlchaos.com](http://www.controlchaos.com)

Dúvidas???



**Até a próxima Aula!!**

