

# Introdução a Métodos Ágeis de Desenvolvimento de Software

Curso de Verão

Centro de Competência em Software Livre

Departamento de Ciência da Computação - IME / USP

Realização: AgilCoop

# Nossa experiência

---

- Cursos de graduação em XP desde 2001
- Apresentações
  - SBES, SUCESU, SEPAI, SEBRAE, etc.
  - Arquivos disponíveis: [www.agilcoop.org.br](http://www.agilcoop.org.br)
- Assessorias em métodos ágeis
- Artigos científicos
- Dissertações de Mestrado

# Roteiro

---

- Motivação
  - Problemas e possíveis direções
- Métodos ágeis
  - Princípios
  - Problemas com os métodos tradicionais
  - Alguns métodos ágeis
- Hoje mais tarde: XP
- Próximos dias: detalhamento

# Novos ventos no mundo do Desenvolvimento de Software

---

## □ Sociedade demanda

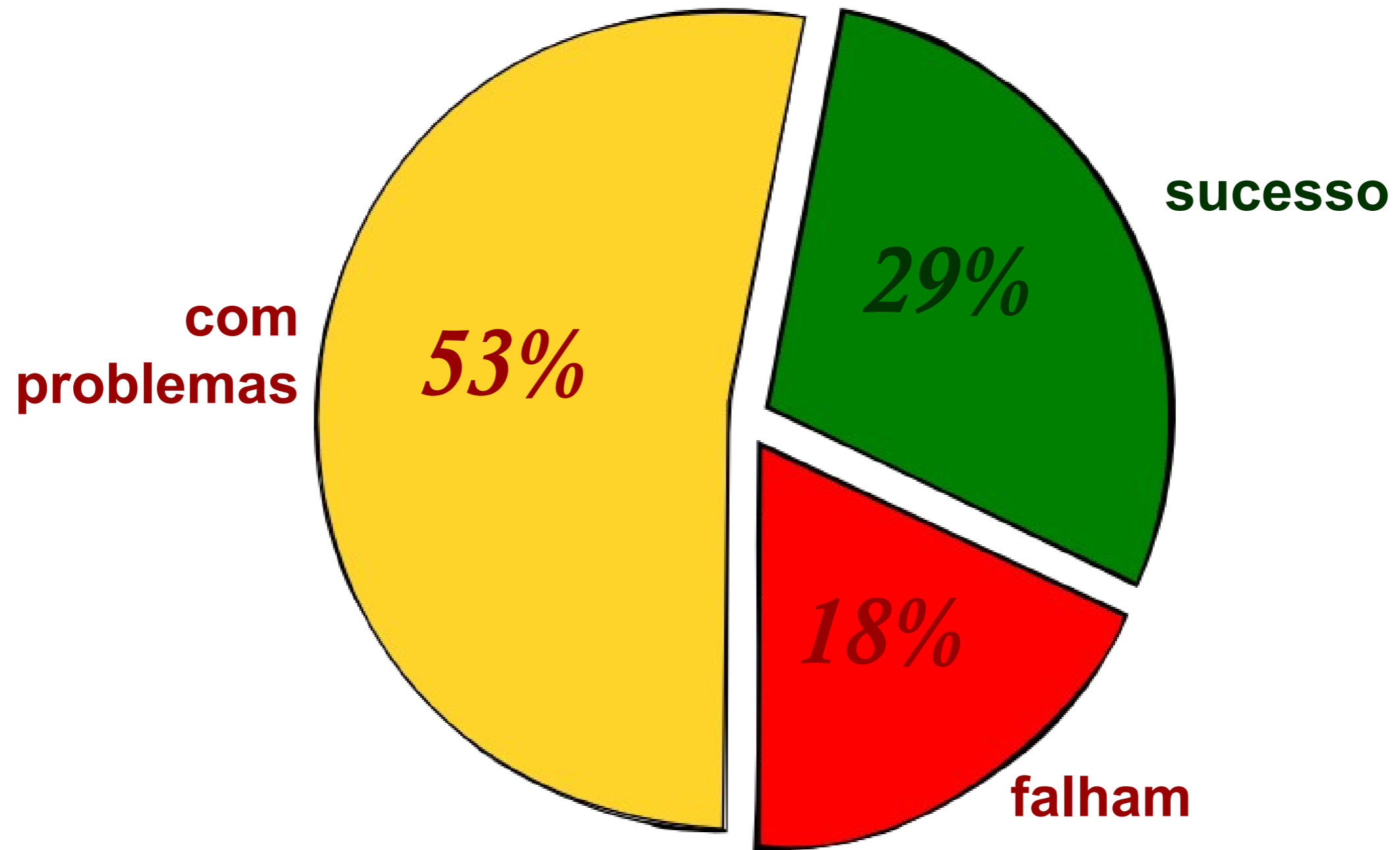
- grande quantidade de sistemas/aplicações
- software complexo, sistemas distribuídos, heterogêneos
- requisitos mutantes (todo ano, todo mês, todo dia)

## □ Mas, infelizmente,

- não há gente suficiente para desenvolver tanto software com qualidade.

# CHAOS Report

□ Resultado dos projetos (2004):



# CHAOS Report

1994

**Projetos** não concluídos  
----- 31%

**Projetos** bem sucedidos  
----- 16%

**Estouro médio de custo**  
-----> 180%

**Estouro médio de prazo**  
-----> 164%

2004

**Projetos** não concluídos  
----- 18%

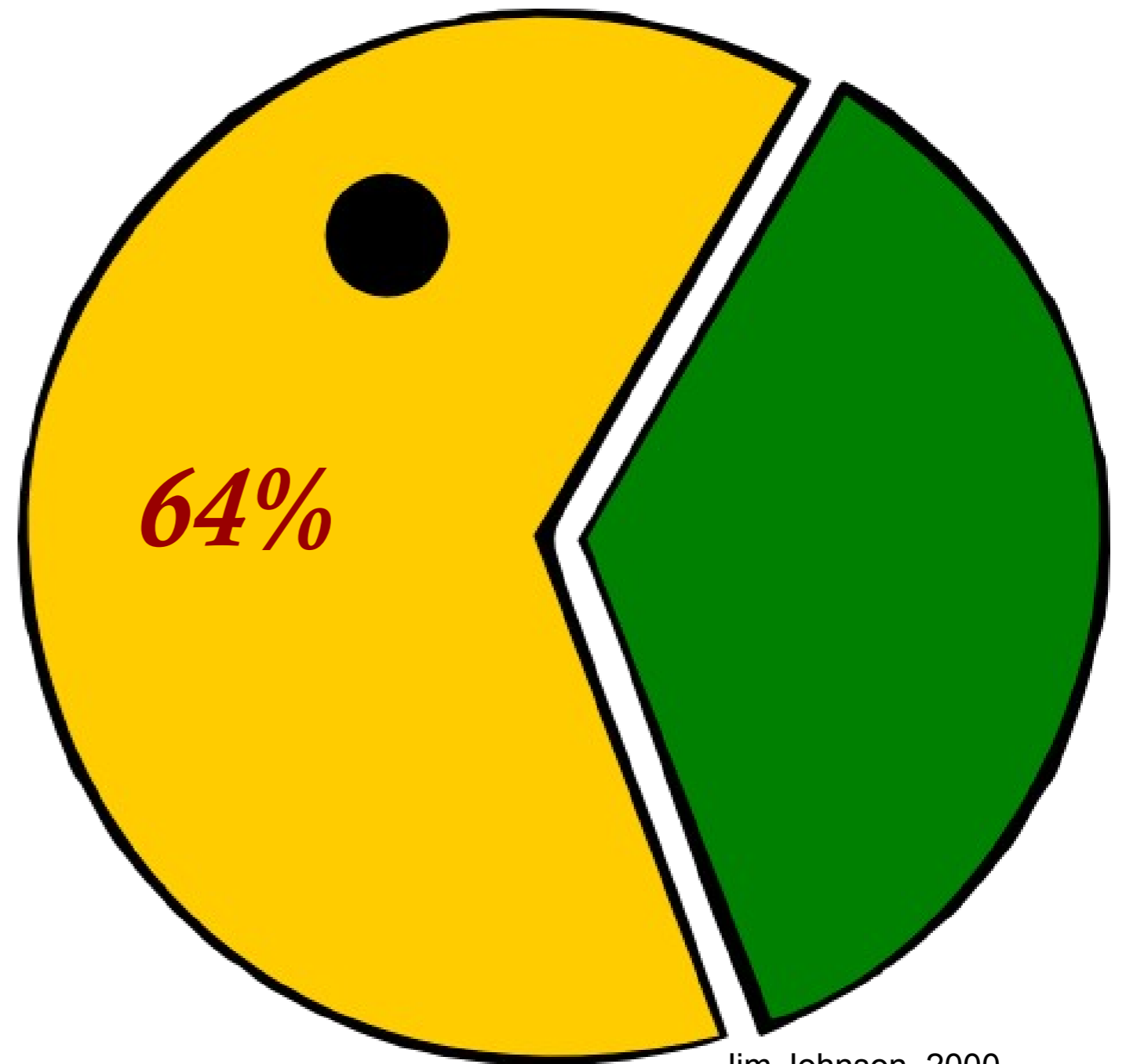
**Projetos** bem sucedidos  
----- 29%

**Estouro médio de custo**  
----- 56%

**Estouro médio de prazo**  
----- 84%

# Qual software?

Funcionalidades  
**nunca** ou  
**raramente**  
utilizadas



Jim Johnson, 2000

# Problemas

---

- Com métodos tradicionais de desenvolvimento
  - Supõem que é possível prever o futuro
  - Pouca interação com os clientes
  - Ênfase em burocracias
    - (documentos, formulários, processos, controles rígidos, etc.)
  - Avaliação do progresso baseado na evolução da burocracia e não do código
- Com software
  - Grande quantidade de erros
  - Falta de flexibilidade

# Como resolver esse impasse?

---

## □ Melhores Tecnologias

- Padrões de Projeto (reutilização de idéias)
- Componentes (reutilização de código)
- Middleware (aumenta a abstração)

## □ Melhores Metodologias

- Métodos Ágeis (o foco deste curso)
- outras... (abordados em outros cursos de ES)

# O que é desenvolvimento de software?

---

## □ Por Alistair Cockburn:

Modelagem (Jacobson)

Engenharia (Meyer)

Disciplina (Humphreys)

Poesia (Cockburn)

Artesanato (Knuth)

Arte (Gabriel)

## □ Erro comum: olhar para software como apenas um desses itens e ignorar os demais

# Jacobson, agosto/2007

BOOK REVIEWS

PRODUCT REVIEWS

EARLIER ISSUES

SEARCH

GO!



[Subscribe to JOT's newsletter](#)

[O-O NEWS & EVENTS](#)

[Previous column](#) [next article](#)

## Enough of Processes - Lets do Practices

REFEREED  
COLUMN



PDF Version

**Ivar Jacobson, Pan Wei Ng and Ian Spence** Ivar Jacobson Consulting

### Abstract

All modern software development processes try to help project teams conduct their work. While there are some important differences between them, the commonalities are far greater - and understandably, since the end goal of them all is to produce working software quickly and effectively. Thus, it doesn't matter which process you adopt as long as it is adaptable, extensible, and capable of absorbing good ideas, even if they arise from other processes.

To achieve this kind of flexibility things need to change. The focus needs to shift from the definition of complete processes to the capture of reusable practices. Teams should be able to mix-and-match practices and ideas from many different sources to create effective ways of working, ones that suit them and address their risks.

# Métodos Ágeis de Desenvolvimento de Software

---

- Movimento iniciado por programadores experientes e consultores em desenvolvimento de software.
- Questionam e se opõem a uma série de mitos/práticas adotadas em abordagens tradicionais de Engenharia de Software e Gerência de Projetos.
- Manifesto Ágil:
  - Assinado por 17 desenvolvedores em Utah em fevereiro/2001.
  - <http://agilemanifesto.org>

# O Manifesto do Desenvolvimento Ágil de Software

---

- **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.

# Princípios do Manifesto Ágil

---

- Objetivo: satisfazer o cliente entregando, rapidamente e com frequência, sistemas com algum valor.
  - Entregar versões funcionais em prazos curtos.
  - Estar preparado para requisitos mutantes.
  - Pessoal de negócios e desenvolvedores juntos.
  - Troca de informações através de conversas diretas.

# Mais princípios do Manifesto Ágil

---

- Medir o progresso através de software funcionando.
- Desenvolvimento sustentável.
- Construir projetos com indivíduos motivados.
- Simplicidade é essencial.
- De tempos em tempos, o time pensa em como se tornar mais eficiente e ajusta o seu comportamento de acordo.

## Em um projeto ágil ideal... (1/2)

- ❑ O gerente de projeto concorda em prosseguir sem que todos os requisitos estejam bem definidos.
- ❑ Os desenvolvedores concordam em prosseguir sem ter todos os requisitos documentados.
- ❑ Os membros da equipe sabem que alguém vai ajudar quando ocorrerem problemas.

## Em um projeto ágil ideal....(2/2)

- Os gerentes percebem que não precisam dizer à equipe o que fazer, ou garantir o que vai ser feito.
- A equipe percebe que ninguém vai dizer o que fazer, isto faz parte do trabalho da equipe.
- Não existem mais a impressão de divisão (*testers and programmers*), todos são desenvolvedores.

# Enquanto isso, no mundo tradicional...

0. Levantamento de Requisitos

1. Análise de Requisitos

2. Desenho da Arquitetura

3. Implementação

4. Testes

5. Produção / Manutenção

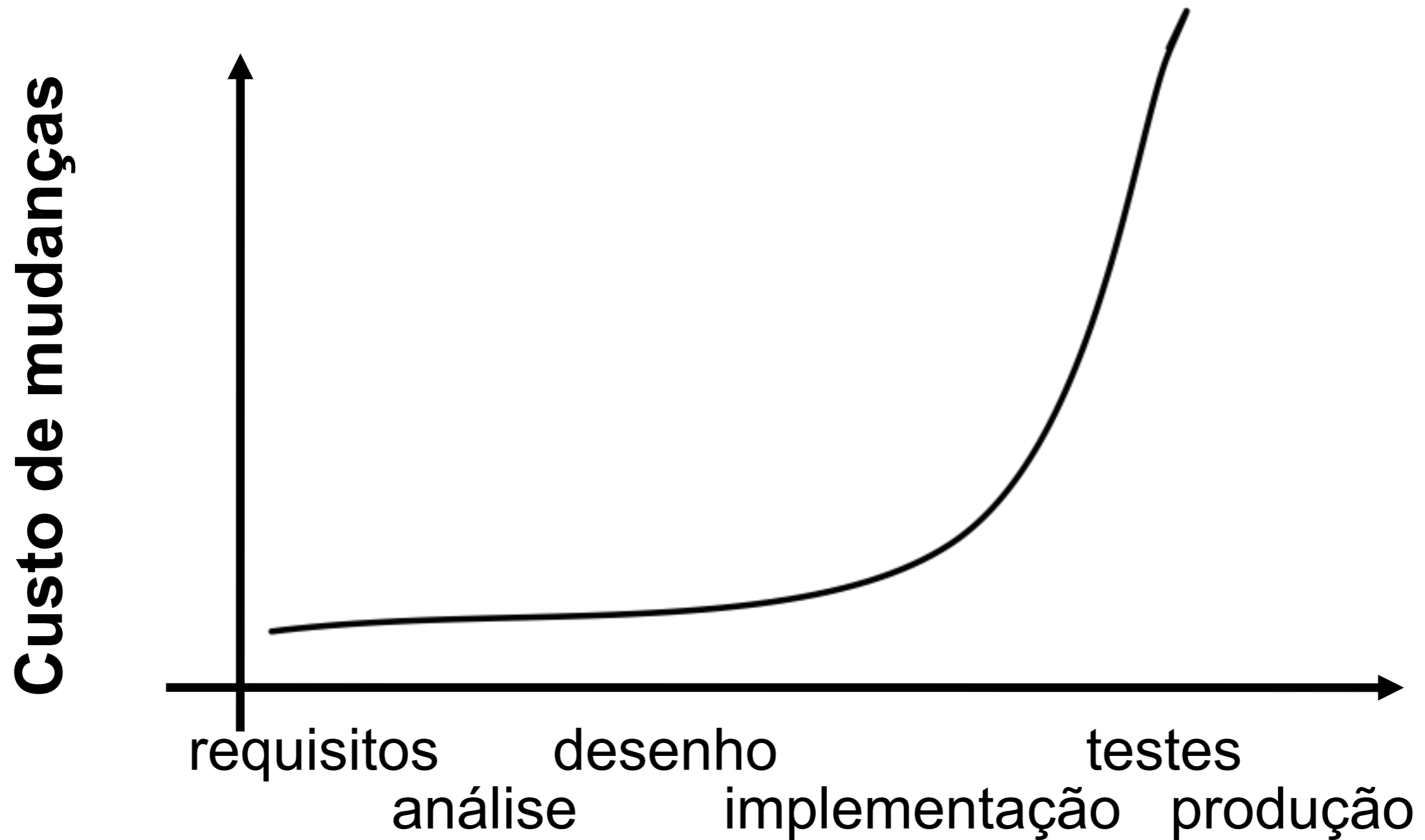
# Premissas Básicas do Modelo Tradicional

---

- É necessário fazer uma análise de requisitos profunda e detalhada antes de projetar a arquitetura do sistema.
- É necessário fazer um estudo minucioso e elaborar uma descrição detalhada da arquitetura antes de começar a implementá-la.
- É necessário testar o sistema completamente antes de mandar a versão final para o cliente.

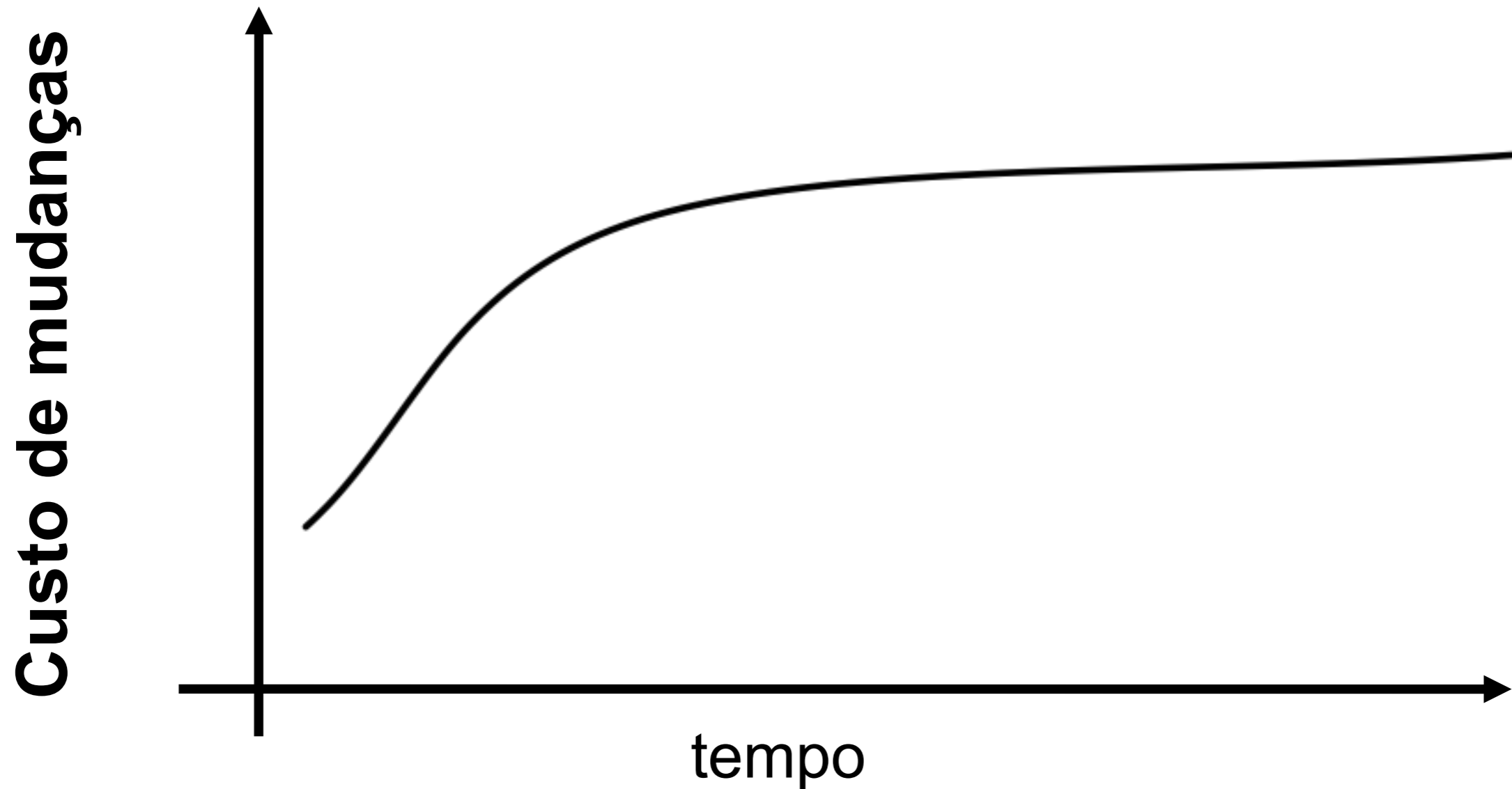
# O que está por trás deste modelo?

---



# E se a realidade hoje em dia fosse outra?

---



# E se essa fosse a realidade?

---

- A atitude dos desenvolvedores de software seria completamente diferente:
  - Tomaríamos as grandes decisões o mais tarde possível.
  - Implementaríamos agora somente o que precisamos *agora*.
  - Não implementaríamos flexibilidade desnecessária (não anteciparíamos necessidades).

# E essa é a nova realidade ! (pelo menos em muitos casos)

---

- **Orientação a Objetos:** facilita e cria oportunidades para mudanças.
- **Técnicas de Refatoração.**
- **Testes automatizados:** nos dão segurança quando fazemos mudanças.
- **Prática / cultura de mudanças:** aprendemos técnicas e adquirimos experiência em lidar com código mutante.

# Principais Métodos Ágeis

---

- Programação eXtrema (XP)
  - O preferido dos desenvolvedores
- Scrum
  - O preferido dos gerentes
- Crystal (uma família de métodos)
- Lean Software Development
- Adaptive Software Development
- Feature Driven Development

# Práticas Ágeis

---

- Planejamento adaptativo (cliente presente)
  - Histórias
  - Detalhamento apenas quando necessário
- Programação em pares (redundância)
- Testes Automatizados (TDD)
- Refatoração (melhoria contínua)
- Pequenos passos
  - ciclo semanal
  - design incremental
  - implantação incremental
- Integração Contínua (repositório único de código)

# Com o código já temos bastante experiência

---

- Programadores se habituam e passam a trabalhar com mais
  - prazer
  - motivação
  - produtividade
  - qualidade
  
- Ferramentas boas para
  - Testes Automatizados / Cobertura
  - Refatoração
  - Coleta de Métricas / Acompanhamento
  - Planejamento e Gerenciamento do Projeto

# Em relação a bancos de dados super-complexos precisamos aprender mais

---

- Evolução de Bancos de Dados muito complexos ainda é difícil.
- Conflitos programadores vs DBAs
- Faltam ferramentas boas para
  - Testes Automatizados
  - Refatoração
  - Coleta de Métricas / Acompanhamento
  - Planejamento e Gerenciamento do Projeto

# Programa deste curso

---

<b>Segunda-feira:</b>	Métodos ágeis (1h50) - Fabio	XP (1h50) - Alfredo
<b>Terça-feira:</b>	Refatoração (1h50) - Mari	Testes (1h50) - Paulo
<b>Quarta-feira:</b>	Planejamento (1h50) - Dairton Scrum (55min) - Dairton	Acompanhamento (55min) - Mari
<b>Quinta-feira:</b>	Lean (55min) - Hugo BDs ágeis (1h50) - Helves	TDD (55min) - Hugo
<b>Sexta-feira:</b>	Dificuldades na implantação (1h50) - Hugo/Dairton Perguntas/Aprofundamento (1h50) - Hugo/Dairton	

# Próximo curso:

# Laboratório de Programação eXtrema

---

**Objetivos:** Através de uma abordagem essencialmente prática, oferecer a oportunidade para desenvolvedores de software e gerentes de TI construírem um pequeno sistema de software de forma colaborativa utilizando XP. O curso será ministrado inteiramente no laboratório Eclipse do IME/USP.

**Ferramentas utilizadas:** No laboratório serão utilizadas

- Java, como linguagem de programação;
- Eclipse, como ambiente de desenvolvimento;
- Subversion, como repositório de código para controle de versões e integração contínua;
- JUnit para testes de unidade;
- Selenium para testes de aceitação e de interface;
- XPlanner, para gerenciamento, acompanhamento e planejamento do desenvolvimento.

# Curso seguinte:

## Desenvolvimento de Software de Qualidade através de Testes Automatizados

---

**Objetivos:** Familiarizar desenvolvedores de software, tanto estudantes quanto profissionais do mercado, com a importância dos testes automatizados e com as principais tecnologias e métodos associados a esta disciplina.

### Conteúdo:

- Importância do Teste de Software.
- Diferença entre testes manuais e automatizados.
- Tipos de testes: testes de unidade, teste de aceitação, teste de estresse, teste de segurança.
- Arcabouços para automação de testes, família xUnit, Selenium, JMeter, etc.
- Cobertura de testes.
- Técnicas avançadas para escritas de bons testes.
- Padrões auxiliares para escrita de testes em sistemas de grande porte e em sistemas com Bancos de Dados.



# Ser Ágil = Vencer medos

- Escrever código
- Mudar de idéia
- Ir em frente sem saber tudo sobre o futuro
- Confiar em outras pessoas
- Mudar a arquitetura de um sistema em funcionamento
- Escrever testes

# Dúvidas?

[agilcoop@agilcoop.org.br](mailto:agilcoop@agilcoop.org.br)



[www.agilcoop.org.br](http://www.agilcoop.org.br)  
(artigos + agilcast)