Laboratório de Engenharia de Software

Custo Total

WAMPS 2015

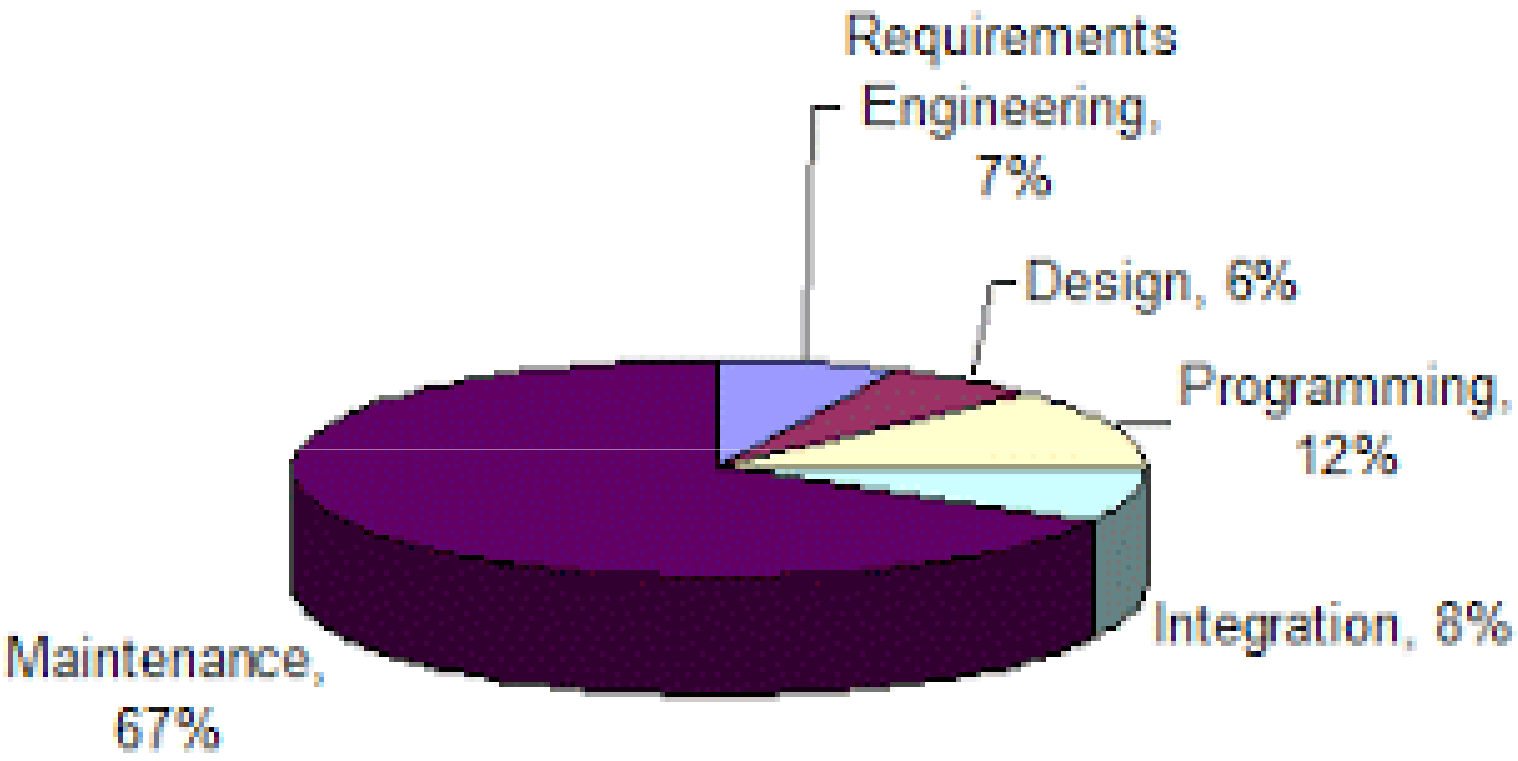
Arndt von Staa

Departamento de Informática

PUC-Rio

Dezembro 2015

Custo do software



Relative Costs of Phases of a Life-cycle [Schach 1999]

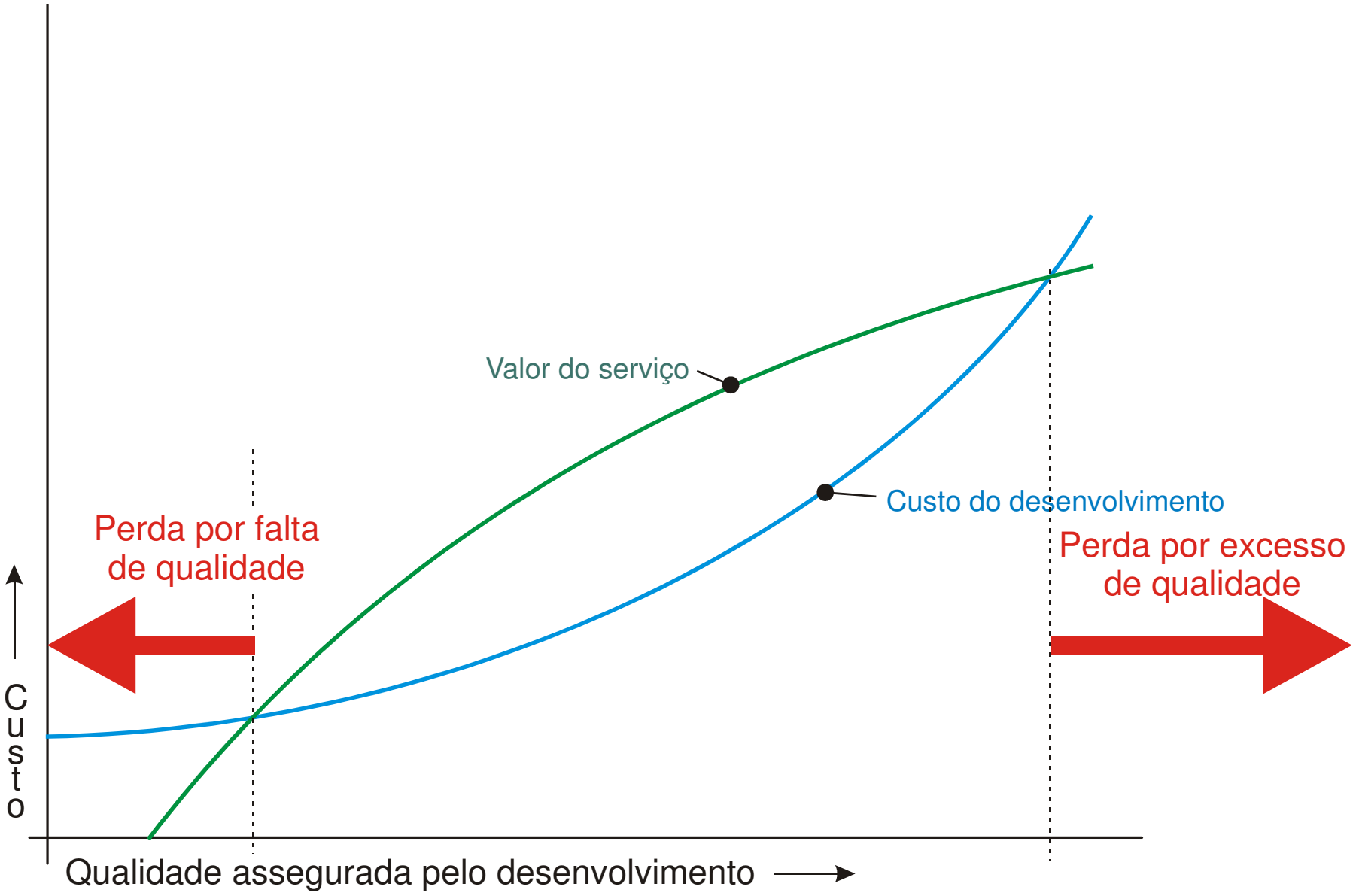
The relative cost for maintaining software and managing its evolution now represents more than 90% of its total cost. (Koskinen, 2010)

Alguns fatores do custo

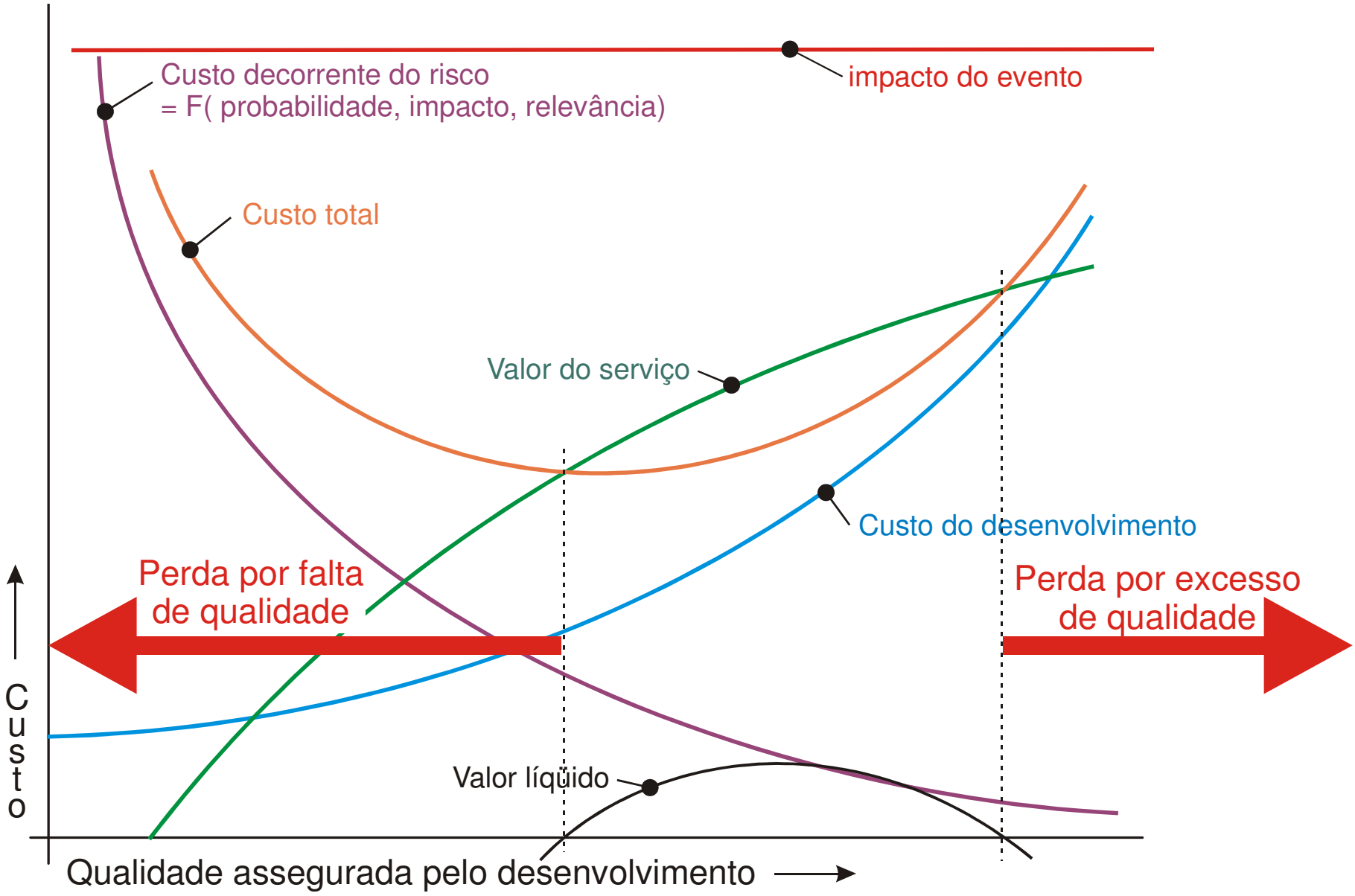
- Desenvolvimento
- Manutenção
  - Corretiva — em torno de 20%
  - Evolutiva — em torno de 55%
  - adaptativa
  - perfectiva

  } em torno de 25%
- Custo das falhas
- Custo do uso
  - usualmente não contabilizado, passado adiante para a vítima: o usuário remoto
- Custo da operação
- Custo do equipamento, rede, nuvem, ...

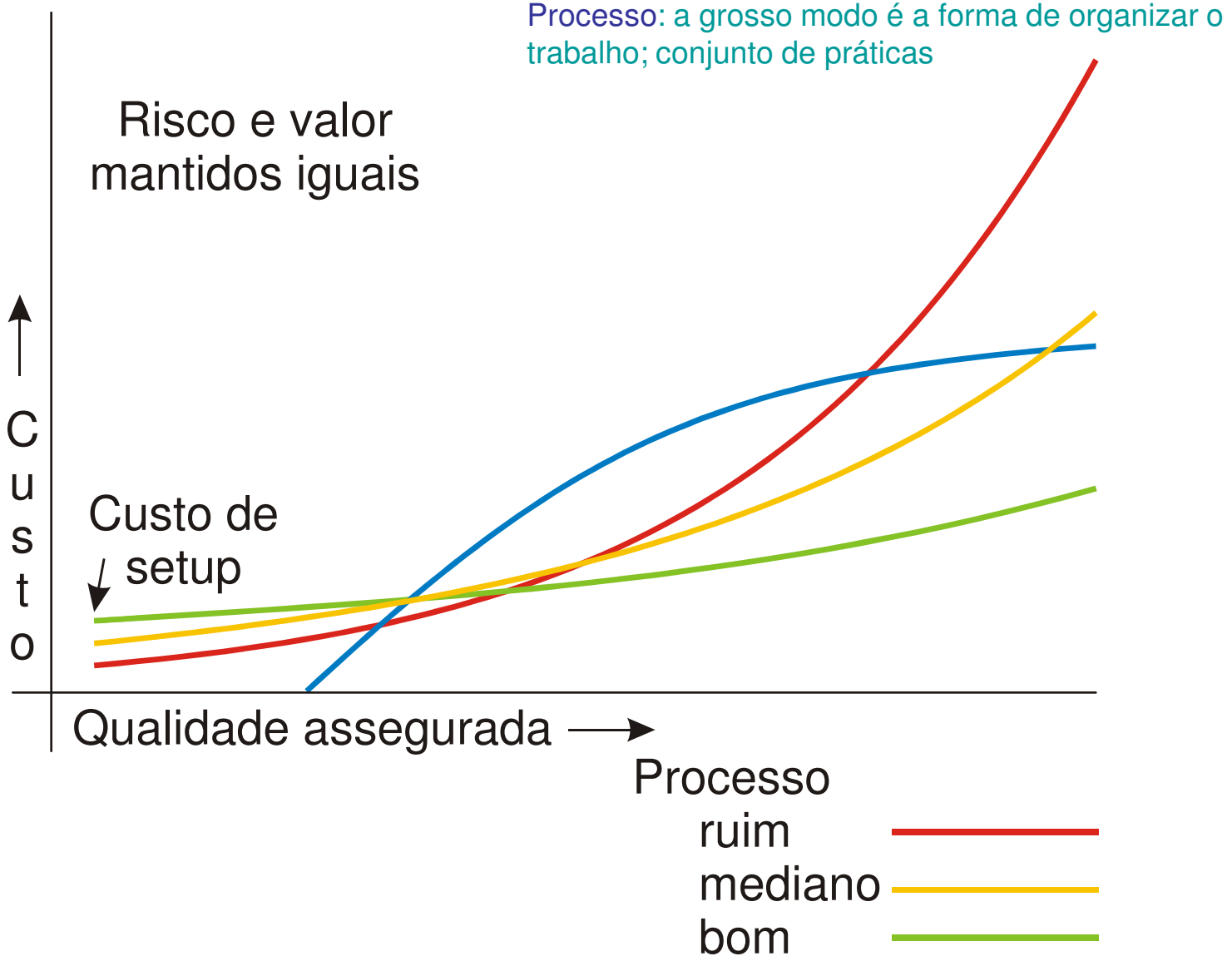
Modelo de economia da qualidade, 1/2



Modelo de economia da qualidade, 2/2



Efeito do processo sobre o custo



Boehm & Basili 2001; IEEE Computer

- Detectar, diagnosticar e remover um defeito de software após entrega é frequentemente 100, ou mais, vezes maior do que detectar e removê-lo durante a especificação, arquitetura e projeto conceitual
  - por temos tantos problemas pós entrega?
  - que tal entender todos os interessados (stakeholders) ?
  - que tal produzir uma arquitetura adequada?
- Projetos atuais dispendem entre 40 e 50% do custo em retrabalho inútil
  - que tal fazer direito de saída?
- Disciplina e boas práticas podem reduzir a taxa de introdução de defeitos em até 75%
  - que tal usar mais técnicas formais leves?

Boehm & Basili 2001; IEEE Computer

Sem alterar nada mais, pode custar até 50% mais para desenvolver software de elevada fidedignidade.

Porém, o investimento tem elevado retorno, se considerarmos custos de operação e manutenção.

Medições pouco formalizadas feitas em empresas de meus alunos:

- usar técnicas formais leves reduz os custos
  - assertivas executáveis
- teste entre 15 e 30%, ao invés de entre 30 e 50%
- defeitos remanescentes perto de zero
- custos devidos a falhas próximo de zero (não ocorreram…)

Como reduzir o custo total?

- Algumas sugestões
  - entender as necessidades de todos os *stakeholders*, mesmo dos ocultos (as vítimas: os usuários)
  - melhorar a proficiência da equipe (MPS.BR-RH?)
    - a variação de produtividade está em torno de 1 para 20
  - usar mais técnicas formais leves
  - fazer direito de saída
    - não usar "faça de qualquer jeito e depois *refatore*" como prática corriqueira
    - minimizar as dívidas técnicas
  - teste automatizado
    - o mais abrangente possível
    - o mais eficaz possível
      - eficácia: número de defeitos observados / número de desfeitos existentes
      - usar mutantes como aproximação de defeitos existentes

Como reduzir o custo total?

- Mais sugestões
  - desenvolver para ser manutenível
    - dispor de um sistema de registro e acompanhamento de incidentes
      - procure implantar ITIL (MPS.BR-SV?) na organização
    - documentação externa
    - reconstrução e reteste automatizados
      - disponibilizar e manter todo o ambiente usado ao desenvolver
      - em especial todos os scripts de teste
    - código bem escrito e formatado
    - facilitar a compreensão dos algoritmos usados
    - facilitar a possibilidade de vir a interoperar
      - especificar com rigor as interfaces

Será que?

- MPS.BR-SW, CMMI-DEV, RUP, ..., efetivamente atingem essas metas?
- Métodos ágeis efetivamente atingem essas metas?
  - OBS. 1 – SCRUM não é um método de desenvolvimento ágil, é um método de gerência ágil
  - OBS. 2 – SCRUM não funciona bem com manutenção, KANBAN funciona melhor
  - OBS. 3 – Cada empresa tem a sua cultura técnica, o processo precisa ser adaptado a ela
  - OBS. 4 – Cada sistema tem necessidades específicas, o processo precisa ser adaptado a elas

Perguntas

Arndt von Staa

arndt@inf.puc-rio.br