



UNIVERSIDADE  
FEDERAL DO CEARÁ  
CAMPUS DE RUSSAS

# Engenharia de Software

## Apresentação da Disciplina

Profa. Dra. Anna Beatriz Marques



# O que é Engenharia de Software?

*“Engenharia de Software é a aplicação de uma abordagem sistemática, disciplinada e quantificável ao desenvolvimento, operação e manutenção de software”*

*IEEE Std 610.12 (1990)*



# Mas eu já sei programar!

» *Por que preciso de Engenharia de Software?*



# Mas eu já sei programar!

## » *Por que preciso de Engenharia de Software?*

- *Programação é parte importante do processo de*
- *Engenharia de Software, mas não é tudo!*


## » *Precisamos também saber...*

- *O que programar,*
- *Como programar,*
- *Se o que foi programado está certo,*
- *etc.*





# Programas acadêmicos

- » *Requisitos estáveis e bem definidos*
  - » *Escopo pequeno (1-10 KLOCS)*
  - » *Prazos razoáveis*
  - » *Equipes pequenas*
  - » *Mão de obra gratuita*
  - » *Não entra em produção*
  - » *Ausência de cliente*
  - » *Ausência de manutenção*
- 

# Programas do “mundo real”

- » *Fazer software no “mundo real” deve considerar fatores como:*
  - ♦ *Custo*
  - ♦ *Prazo*
  - ♦ *Qualidade*
- » *Em função do tamanho do software, esses fatores se tornam difíceis de garantir!*



# Cenário 1: Agenda

- » *Objetivo*
  - *Guardar eventos, reuniões e compromissos.*
  
- » *Quanto custa para fazer?*
- » *Quanto tempo vai levar para ficar pronto?*
- » *Qual a consequência no caso de defeito?*



## Cenário 2: Boeing 777

- » *Objetivo*
  - ◇ *Controlar todo o hardware do Boeing 777*

- » *Quanto custa para fazer?*
- » *Quanto tempo vai levar para ficar pronto?*
- » *Qual a consequência no caso de defeito?*





# O que acontece se o desenvolvedor errar?

» CASO REAL 1: Therac - 25

## **Máquina de radioterapia controlada por computador**

- **Problema:**
  - Doses indevidas de radiação emitidas
- **Causa:**
  - Interface com usuário inapropriada
  - Documentação deficiente
  - Software reutilizado sem ser adaptado para o novo hardware
  - Software de sensores de falha com defeito
- **Consequências**
  - Ao menos 5 mortes entre 1985 e 1987

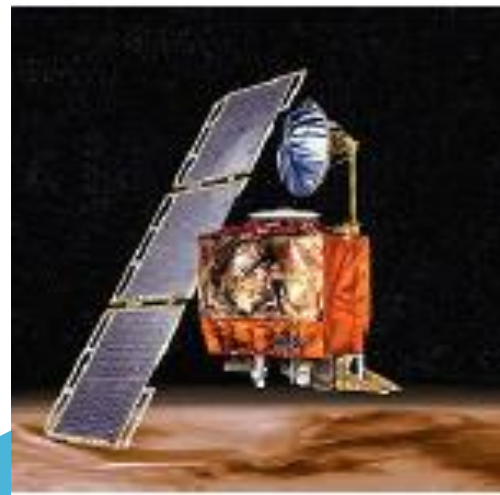


# O que acontece se o desenvolvedor errar?

» CASO REAL 2: Mars Climate Orbiter


**Sonda da NASA lançada para realizar o estudo das variáveis atmosféricas**

- **Problema:**
  - Foi destruído devido a um erro de navegação
- **Causa:**
  - A equipe da Terra usou medidas inglesas para calcular os parâmetros de inserção e enviou os dados a nave e esta apenas realizavam cálculos no sistema métrico.
- **Consequências**
  - perda de US\$ 300 milhões



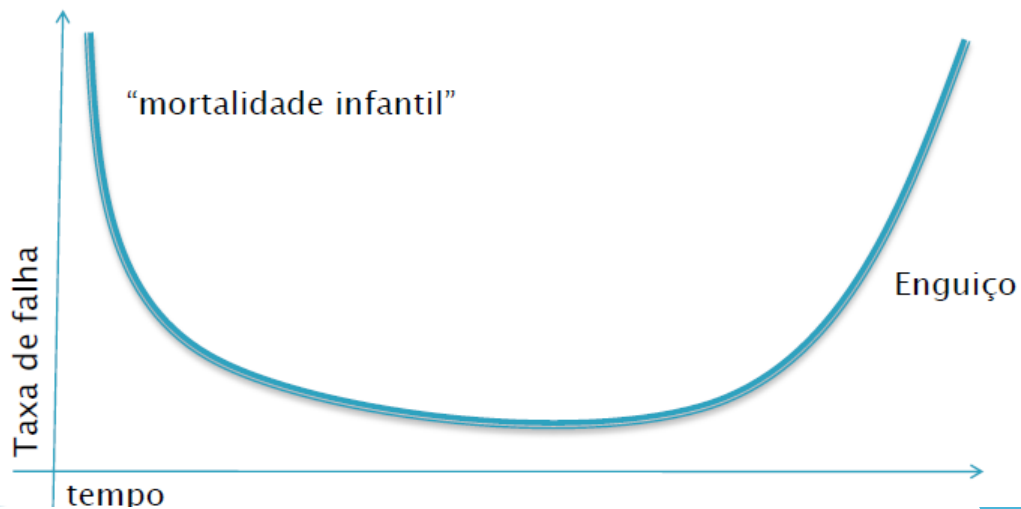


# Software X Hardware

- » Software é desenvolvido por um processo de engenharia
    - ◇ Alto custo de criação (na maioria desenvolvido sob encomenda)
    - ◇ Baixo custo de reprodução
    - ◇ Não se desgasta
    - ◇ Defeitos no produto usualmente são consequências de problemas no processo de desenvolvimento.
  - » Hardware é manufaturado
    - ◇ Alto custo de reprodução
    - ◇ Pode desgastar
    - ◇ Pode ser substituído na totalidade ou em partes
- 

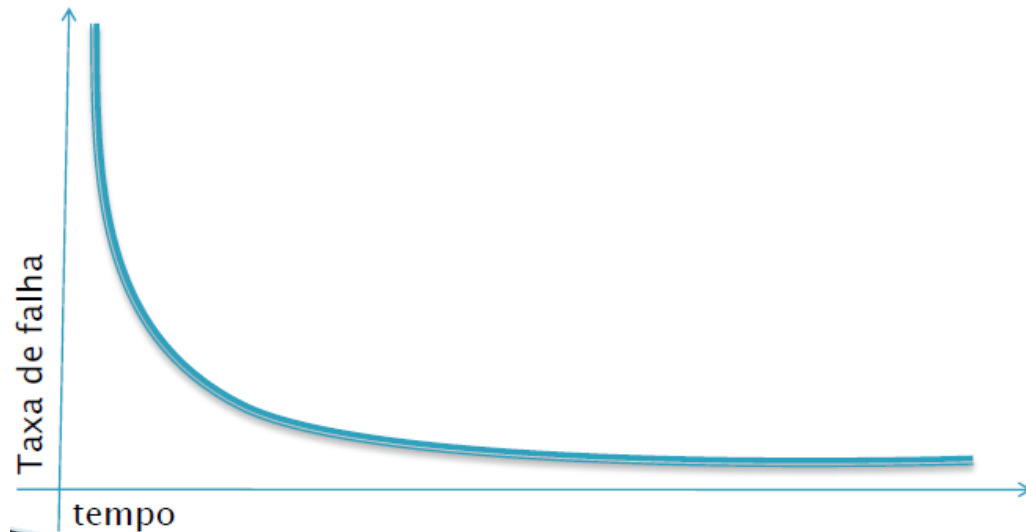
# Software X Hardware

» Curva de falha de hardware



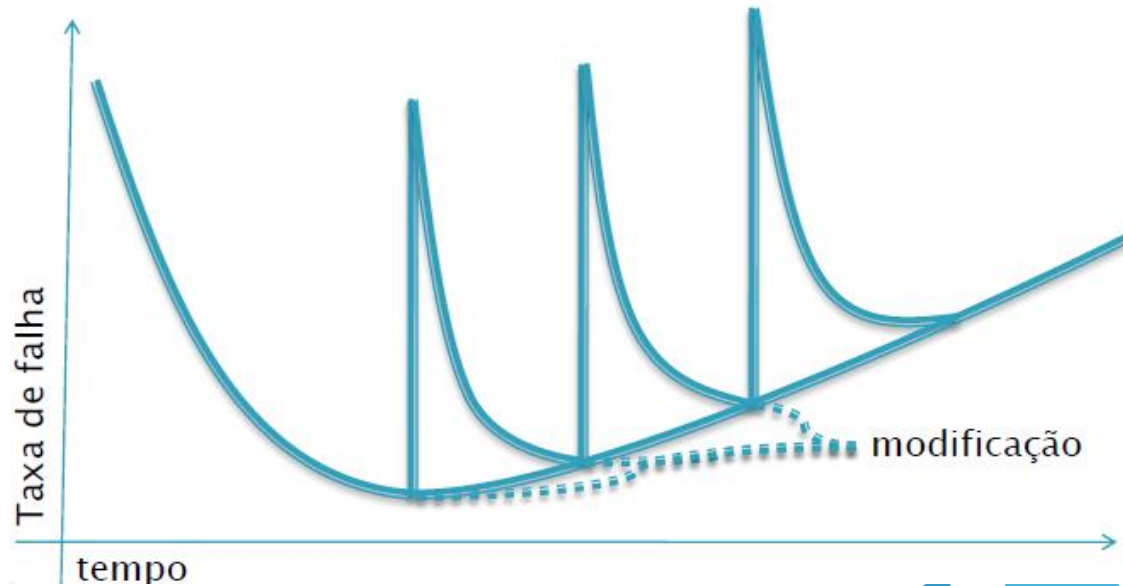
# Software X Hardware

» Curva **ideal** de falha de software



# Software X Hardware


» Curva **real** de falha de software





# Atributos de um bom software

Característica do produto	Descrição
Manutenabilidade	O software deve ser construído de tal forma que possa atender as novas necessidades do cliente. A mudança é uma exigência inevitável em um ambiente de negócios em constante mudança.
Confiabilidade e segurança	Software confiáveis não devem causar danos físicos ou econômicos em caso de falha no sistema. Usuários com más intenções não devem ser capazes de acessar ou danificar o sistema.
Eficiência	Um software eficiente não deve desperdiçar recursos do sistema, como memória ou processador. Eficiência inclui a capacidade de resposta, tempo de processamento, utilização de memória, etc.
Aceitabilidade	Software deve ser aceitável pelos usuários para os quais foi projetado. Isto significa que o software deve ser compreensível, usável e compatível com outros sistemas que estes usuários utilizam.




# Elementos da Engenharia de Software







# Elementos da Engenharia de Software

- » A base em que se apoia é o foco em qualidade
  - » Processo:
    - ◇ A camada de processo é o alicerce
    - ◇ Define os passos gerais para o desenvolvimento e manutenção do software
    - ◇ Serve como uma estrutura de encadeamento de métodos e ferramentas
  - » Métodos
    - ◇ São os “how to’s” de como fazer um passo específico do processo para construir software
  - » Ferramentas
    - ◇ Automatizam o processo e os métodos.
- 

# Receita de brigadeiro

1. Coloque em uma panela funda o leite condensado, a margarina e o chocolate em pó.
2. Cozinhe [no fogão] em fogo médio e mexa sem parar uma colher de pau.
3. Cozinhe até que o brigadeiro comece a desgrudar da panela.
4. Deixe esfriar bem, então unte as mãos com margarina, faça as bolinhas e envolva-as em chocolate granulado.



O que é processo,  
método e  
ferramenta?



# Receita de brigadeiro



1. **Coloque** em uma **panela** funda o leite condensado, a margarina e o chocolate em pó.
2. **Cozinhe** [no **fogão**] em fogo médio e **mexa** sem parar com uma **colher de pau**.
3. **Cozinhe** até que o brigadeiro comece a desgrudar da **panela**.
4. **Deixe esfriar bem**, então **unte as mãos** com margarina, **faça** as bolinhas e **envolva**-as em chocolate granulado.

Processo

Ferramentas

Métodos

# Mitos do Software

- » Basta um bom livro de Engenharia de Software para fazer bom software.

**Realidade:**

Um bom livro certamente ajuda, mas ele precisa refletir as técnicas mais modernas de Engenharia de Software e ser lido!

# Mitos do Software

- » Se estivermos com o cronograma atrasado, basta adicionar mais gente ao projeto

## **Realidade:**

- Adicionar gente a um projeto atrasado faz o projeto atrasar mais!
- As pessoas que estão entrando terão que aprender sobre o projeto antes de começar a ajudar no desenvolvimento.
- As pessoas que estão no desenvolvimento, terão que parar para explicar aos que estão entrando

# Mitos do Software

- » Se o projeto for terceirizado, todos os meus problemas estão resolvidos.

**Realidade:**

É mais difícil gerenciar projetos terceirizados do que projetos internos!

# Mitos do Software

- » Basta dar uma ideia geral do que é necessário no início

## **Realidade:**

- Requisitos ambíguos normalmente são uma receita para desastre!
- Comunicação contínua com o cliente é fundamental!

# Mitos do Software

- » Modificações podem ser facilmente acomodadas, porque software é flexível.

## **Realidade:**

- O impacto de modificações no software varia em função da modificação e do momento em que ela é requisitada!
- Comunicação contínua com o cliente é fundamental!



# Mitos do Software

- » Assim que o código for escrito o trabalho termina.

## **Realidade:**

- 60% a 80% do esforço será gasto depois que o código foi escrito!
- Vale a pena esforçar para chegar a um bom código (boa documentação, bom projeto, etc.)!

# Mitos do Software

- » Só é possível verificar a qualidade de um software quando o executável existir

**Realidade:**

Revisões usualmente são mais eficazes que testes, e podem ser utilizadas antes do software estar executável!

# Mitos do Software

- » O único produto a ser entregue em um projeto é o código.

**Realidade:**

Além do código, documentações tanto para a manutenção quanto para o uso são fundamentais!

# Mitos do Software

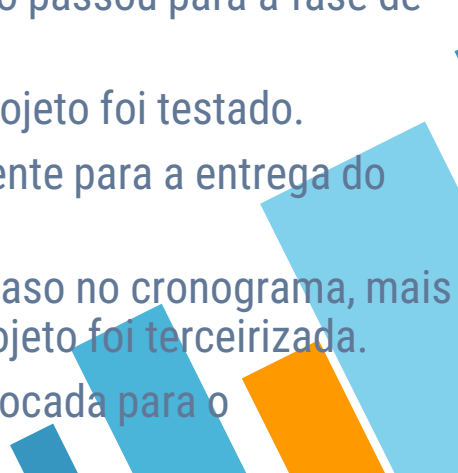
- » Engenharia de software gera documentação desnecessária.

## **Realidade:**

- Engenharia de software foca em criar qualidade, e não criar documentos!
- Algum grau de documentação é necessário para evitar retrabalho!
- Questione sempre que encontrar um documento desnecessário para o projeto!



# Jogo dos “sete” erros

- » A nossa empresa fez o levantamento dos requisitos com o cliente tentando esclarecer todas as ambiguidades.
  - » Após a fase de levantamento dos requisitos, o projeto passou para a fase de codificação.
  - » Ao final da codificação e geração do executável, o projeto foi testado.
  - » Só após o teste, a empresa acionou o cliente novamente para a entrega do código gerado.
  - » Durante a fase de codificação e após verificar um atraso no cronograma, mais profissionais foram incluídos na equipe e parte do projeto foi terceirizada.
  - » Após a codificação do produto, toda a equipe foi deslocada para o desenvolvimento de outro projeto.
- 

# Nós precisamos de Engenharia de Software



Como o cliente explicou



Como o líder de projeto entendeu



Como o analista planejou



Como o programador codificou



O que os beta testers receberam



Como o consultor de negócios descreveu



Valor que o cliente pagou



Como o projeto foi documentado



O que a assistência técnica instalou



Como foi suportado



Quando foi entregue



O que o cliente realmente necessitava

## Objetivo da disciplina

- » Introduzir através do estudo dos conceitos de engenharia de software os conhecimentos básicos necessários para entender o funcionamento dos processos de software, elicitação de requisitos e gerência de projetos.

## Conteúdos a serem abordados

- » Visão geral e introdutória dos princípios fundamentais e éticos-profissionais da Engenharia de Software. Introdução às atividades de engenharia de requisitos; projeto de software; modelos de desenvolvimento; e conceitos de gerenciamento de projetos (qualidade, estimativa de custo, etc.) na engenharia de software



## Objetivos de aprendizagem

- » (i) fornecer uma visão geral de Engenharia de Software
- » (ii) levantar requisitos e validá-los;
- » (iii) elaborar um documento de requisitos;
- » (iv) explicar, analisar e modelar processos de desenvolvimento de software;
- » (v) fornecer conceitos básicos de gerência de projetos; e
- » (vi) listar possíveis documentos utilizados em um processo de software.

# Qual metodologia de ensino a ser adotada?

- » Aulas expositivas
- » Seminário de artigos científicos
- » Trabalho Prático
- » Roteiros de estudo com gamificação
- » Atividades gamificadas

# Como o Aprendizado será avaliado?

- » Frequência  $\geq 75\%$ 
  - ◇ Realização de atividades síncronas e assíncronas
- » Avaliações Progressivas (APs)
  - ◇ Provas Parciais
  - ◇ Seminário
  - ◇ Trabalho Prático
  - ◇ Atividades práticas sobre os conteúdos abordados



# Bibliografia recomendada

- **Bibliografia Básica:**
  1. SOMMERVILLE, I. Engenharia de software. 9 ed. Addison Wesley, 2011. ISBN: 9788579361081.
  2. PRESSMAN, R. Engenharia de software. 6 ed. Pearson, 2009.
  3. LARMAN, Craig. Utilizando UML e padroes : uma introducao a analise e ao projeto orientados a objetos. 5. ed. Porto Alegre: Bookman, 2007. 695 p. ISBN 856003152-9.



**OBRIGADA!**

