Engenharia de Software

Natália Schots

Aula 2

Agenda

- Introdução à Engenharia de Software
  - Elementos básicos
  - Histórico
  - Mitos da Engenharia de Software
  - Princípios de Hooker

Introdução à ES

O que é Engenharia de Software?

- "O estabelecimento e uso de um conjunto de **princípios de engenharia** com o objetivo de se construir software confiável, eficiente e viável economicamente em máquinas reais" (F. L. Bauer, 1969)

- "Aplicação de uma **abordagem sistemática, disciplinada e quantificativa** para o desenvolvimento, operação e manutenção de software, isto é, a aplicação da engenharia ao software" (IEEE, 1993)

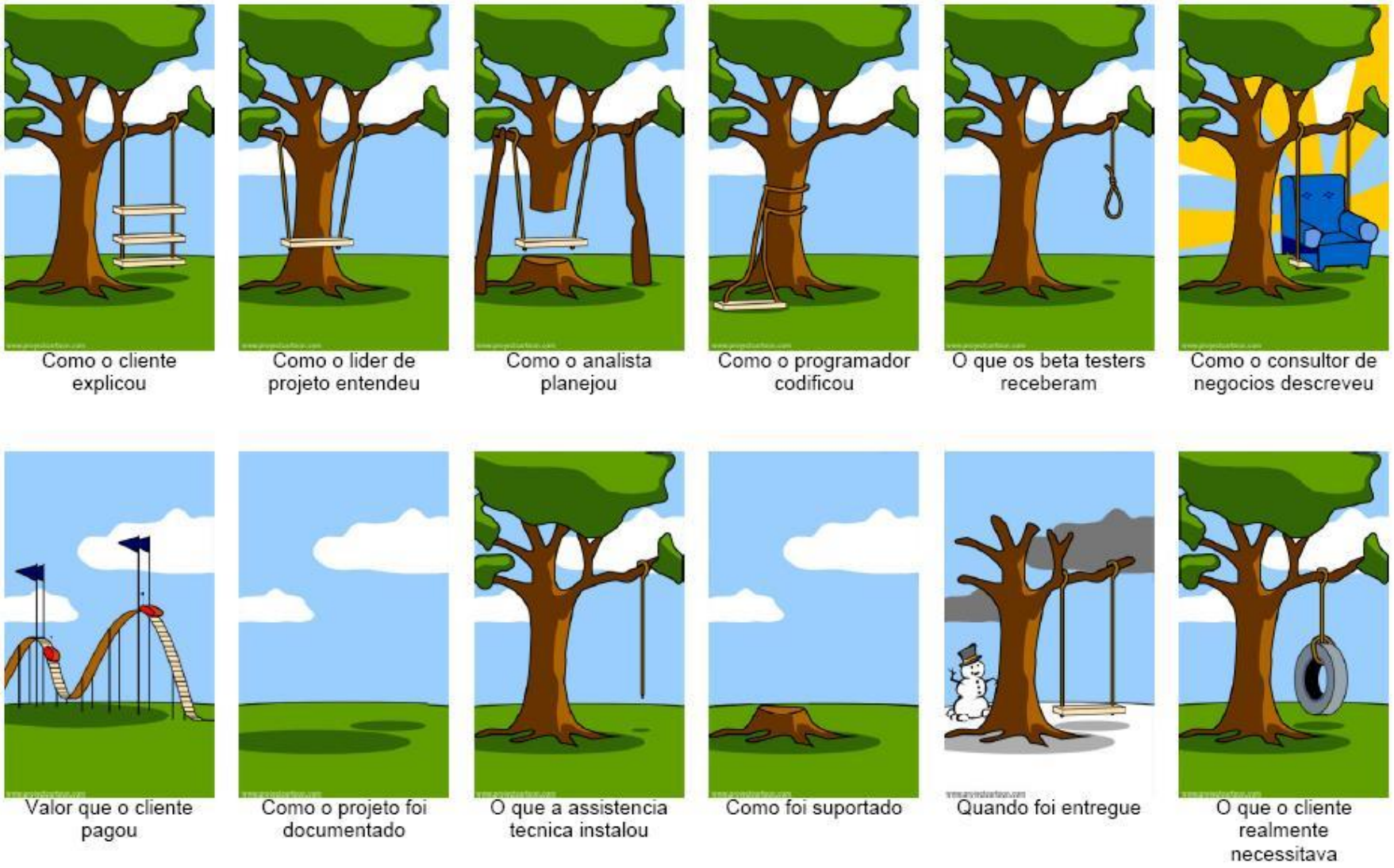
Engenharia no geral (1/2)

- Busca a **resolução de problemas práticos** por meio de soluções que sejam economicamente viáveis
  - Motivada pela limitação de recursos, tais como: tempo, dinheiro e pessoal capacitado
- A estratégia da engenharia é **aplicar conhecimento científico** sobre um determinado domínio tecnológico
- A engenharia se especializa em diversos ramos de conhecimento: elétrica, civil, software...

Engenharia no geral (2/2)

- Etapas na Engenharia:
  - Análise
    - O problema deve ser dividido em partes menores e mais simples até que estas partes possam ser resolvidas
  - Síntese
    - Unir as soluções de cada parte em uma estrutura maior que atenda todo o problema
  - Correções
    - Resolução de problemas decorrentes da tradução durante a síntese (verificação) ou de elicitação durante a análise (validação)

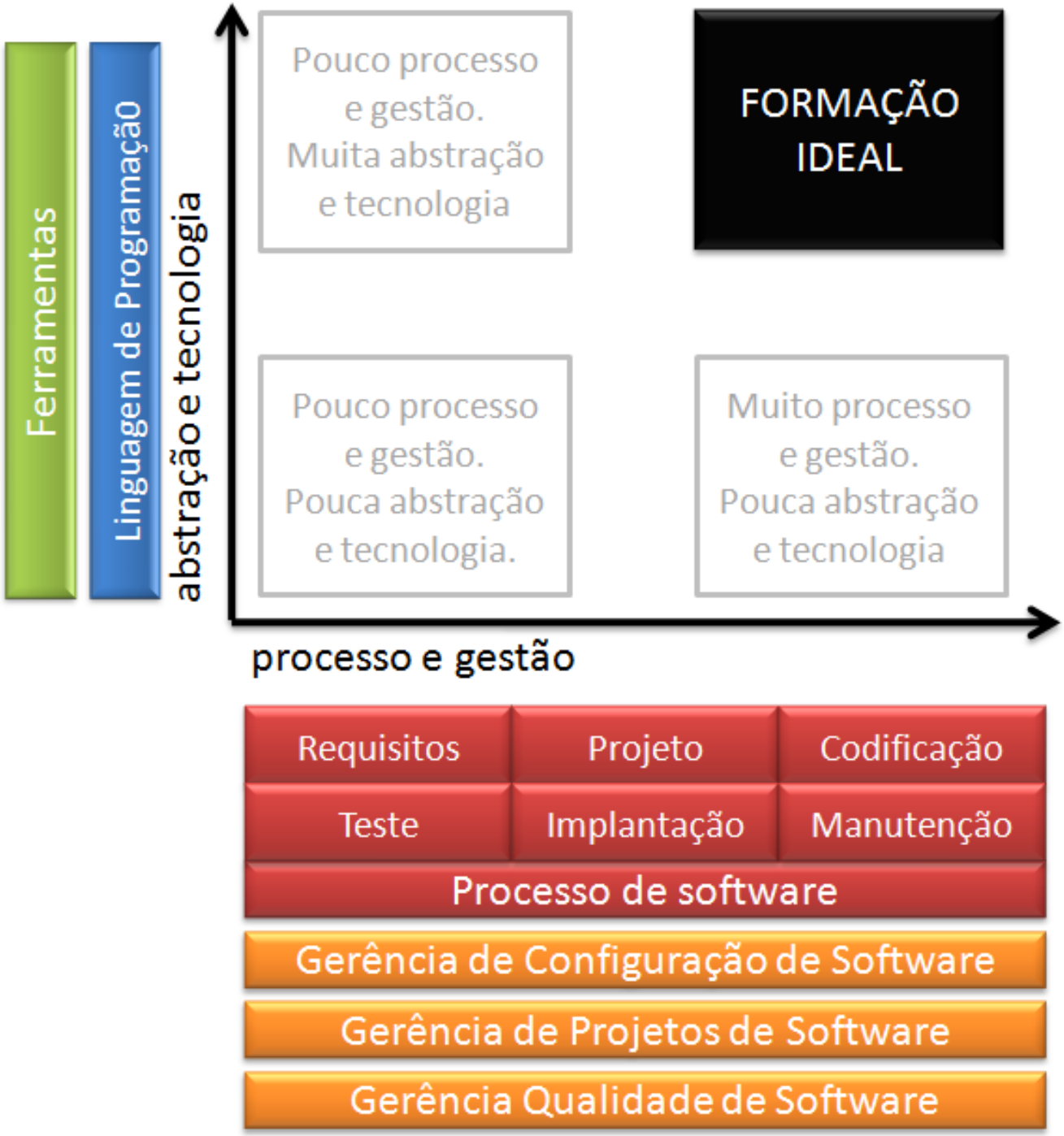
Por que Engenharia de Software? (1/2)



Por que Engenharia de Software? (2/2)

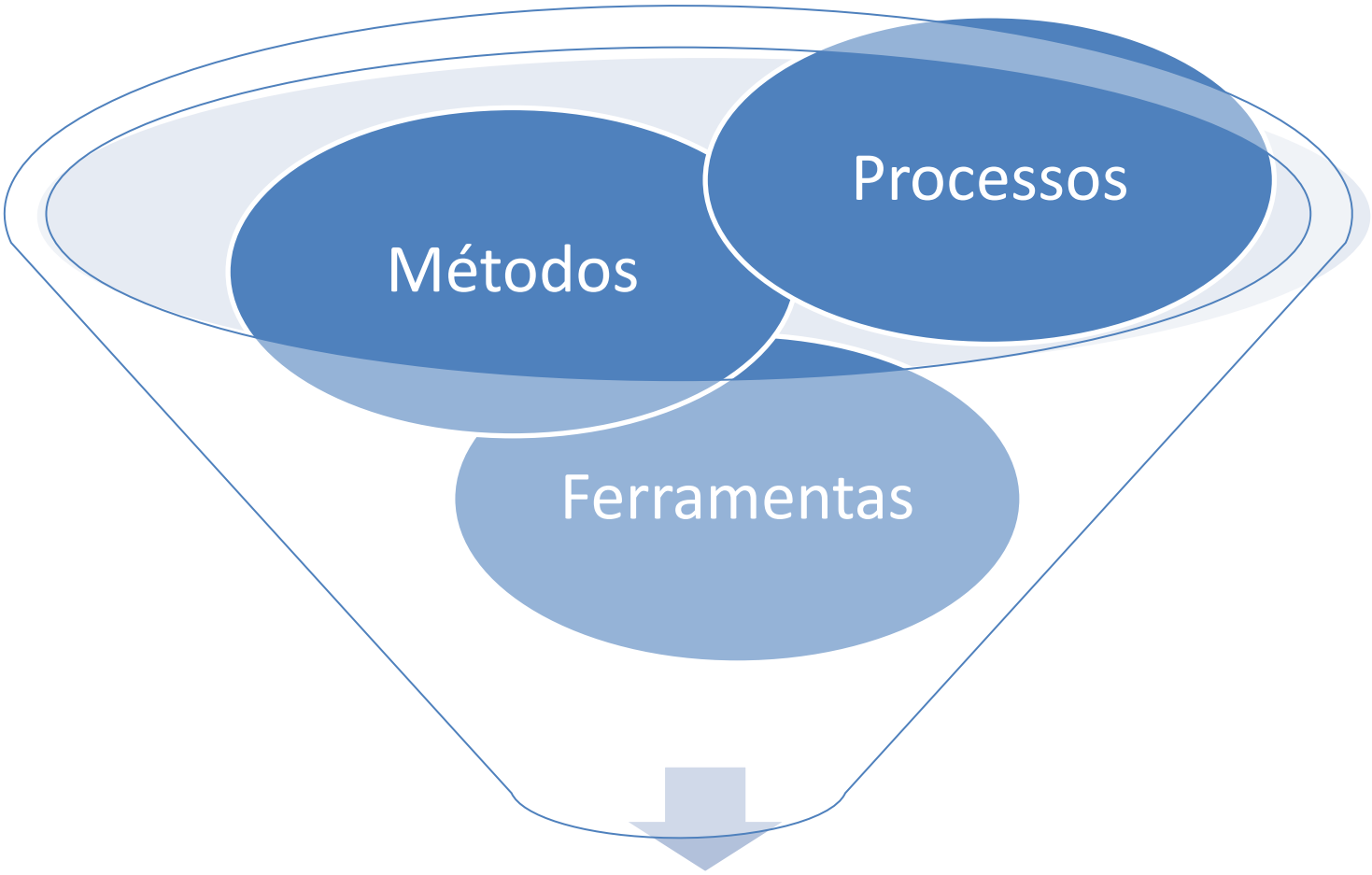
- Além de saber programar, é necessário saber:
  - O **que** programar
  - **Como** programar
  - Se o que foi programado está **certo**?
  - …

- É necessário conhecer o processo de desenvolvimento e seus produtos, saber medi-los e melhorá-los continuamente


Ferramentas
Linguagem de Programação
abstração e tecnologia
Pouco processo e gestão. Muita abstração e tecnologia
FORMAÇÃO IDEAL
Pouco processo e gestão. Pouca abstração e tecnologia.
Muito processo e gestão. Pouca abstração e tecnologia
processo e gestão
Requisitos
Projeto
Codificação
Teste
Implantação
Manutenção
Processo de software
Gerência de Configuração de Software
Gerência de Projetos de Software
Gerência Qualidade de Software

Elementos básicos

O que compõe a ES?



Processos

- Definem os passos gerais para o desenvolvimento e manutenção do software
- Servem como uma estrutura de encadeamento de métodos e ferramentas

Métodos

- Descrevem como fazer um passo específico do processo

- Representação do software durante seu desenvolvimento
  - Notação e linguagens

Ferramentas

- Automatizam o processo e os métodos
  - Ferramentas CASE (*Computer Aided Software Engineering*)
  - Ambientes de desenvolvimento de software – IDEs
  - …
- Cuidado com o "desenvolvimento guiado por ferramentas"
  - É importante usar a ferramenta certa para o problema
  - O problema não deve ser adaptado para a ferramenta disponível

Processos, métodos ou ferramentas?

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

Processos, métodos ou ferramentas?

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

Normal: processo
Em negrito: método
Em itálico: ferramenta

Qual a exigência de cada elemento?

- Depende do contexto da organização

- Pensar nos elementos como “pernas” de um banco
  - O tamanho de cada “perna” dependerá do solo da organização

“Não existe bala de prata!”

- O processo, os métodos e as ferramentas devem ser escolhidos **em função do problema e seu contexto**
- Cuidado!
  - Exigir menos do que o necessário pode levar a desordem
  - Exigir mais do que o necessário pode emperrar o projeto
- Processos devem ser adaptados para o contexto real de uso
  - Características do projeto
  - Características da equipe
  - Características do cliente

Histórico

Antes da Engenharia de Software...

- A programação era considerada uma espécie de **arte**
- Os **altos custos de hardware** escondiam os custos de software
- Os sistemas eram **simples** e construídos por pequenas equipes
- Os sistemas eram construídos para resolver **problemas específicos**

Crise do Software

- Teve início em meados da década de 1960
- Os custos de hardware começaram a se reduzir
- Os computadores se tornaram cada vez mais velozes
- A capacidade de armazenamento aumentou
- A demanda por software cresceu
- As equipes de desenvolvimento não acompanharam a demanda por produção

Origens da ES

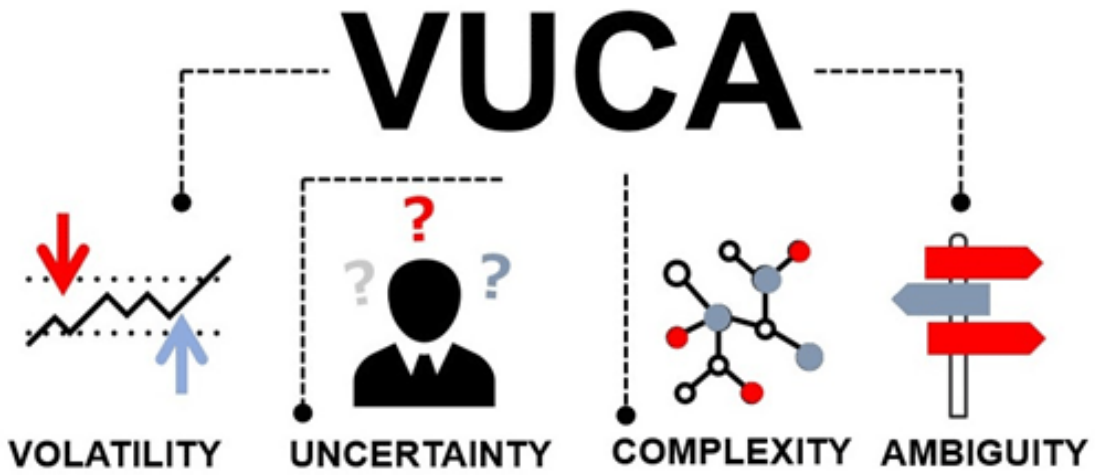
- A Engenharia de Software surgiu com o intuito de:
  - Identificar e analisar as causas dos problemas envolvidos com o desenvolvimento de software
  - Propor **soluções economicamente viáveis** para a resolução destes problemas
  - **Organizar o conhecimento** sobre técnicas disponíveis para o desenvolvimento de software

Depois da Engenharia de Software...

- 1970s:
  - *Lower-CASE tools* (programação, depuração, colaboração)
  - Ciclo de vida cascata
  - Desenvolvimento estruturado
- 1980s:
  - Ciclo de vida espiral
  - Desenvolvimento orientado a objetos
- 1990s:
  - *Upper-CASE tools*
  - Processos
  - Modelagem
- Atualmente:
  - Métodos ágeis
  - Desenvolvimento dirigido por modelos
  - Linhas de produto
  - Experimentação
  - Desenvolvimento distribuído

Atualmente

- Vivemos em um mundo V.U.C.A.



- Necessidade de novas abordagens a cada instante!

Mitos da Engenharia de Software

O que são?

- É comum a criação e disseminação de certos mitos em Engenharia de Software (ES)
  - Muitos deles possuem aspecto intuitivo, baseados em fatos razoáveis, o que facilita sua disseminação

- A fonte destes mitos é normalmente devido a:
  - Más experiências
  - Desconhecimento da teoria da disciplina de ES aplicada na prática

Mitos gerenciais

- “Basta um bom livro de ES para fazer bom software”
  - Um bom livro certamente ajuda, mas ele precisa refletir as técnicas mais modernas de ES (e ser lido)
- “Se estivermos com o cronograma atrasado, basta adicionar mais gente ao projeto”
  - Adicionar gente a um projeto atrasado normalmente faz o projeto atrasar ainda mais!
- “Se o projeto for terceirizado, todos os meus problemas estão resolvidos”
  - É mais difícil gerenciar projetos terceirizados do que projetos internos

Mitos do cliente

- “Basta dar uma ideia geral do que é necessário no início”
  - Requisitos ambíguos normalmente são uma receita para um desastre
  - Comunicação contínua com o cliente é fundamental

- “Modificações podem ser facilmente acomodadas, porque software é flexível”
  - O impacto de modificações no software varia em função da modificação e do momento em que ela é requisitada

Mitos do desenvolvedor (1/2)

- “Assim que o código for escrito, o trabalho termina”
  - 60% a 80% do esforço será gasto depois que o código foi escrito
  - Vale a pena se esforçar para chegar a um bom código (boa documentação, bom projeto etc.)

- “Só é possível verificar a qualidade de um software quando o executável existir”
  - Revisões usualmente são mais eficazes que testes, e podem ser utilizadas antes do software estar executável

Mitos do desenvolvedor (2/2)

- “O único produto a ser entregue em um projeto é o código”
  - Além do código, documentações tanto para a manutenção quanto para o uso são fundamentais
- “Engenharia de software gera documentação desnecessária”
  - ES 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

Princípios de Hooker

Princípio #1

- Tem que existir uma razão para se fazer software
  - Se não for possível identificar essa razão, é melhor não fazer
  - Fazer software, em última instância, consiste em **agregar valor para o usuário**
  - É importante enxergar os reais requisitos do software!

Princípio #2

- Keep it simple, sir! (KISS)
  - "um projeto deve ser o mais simples possível, mas não mais simples que isso"
  - **As soluções mais elegantes normalmente são simples**
  - Fazer algo simples usualmente demanda mais tempo do que fazer de forma complexa

Princípio #3

- Mantenha o estilo
  - O projeto de um software deve **seguir um único estilo**
  - A combinação de diferentes estilos corretos pode levar a um software incorreto
  - Padrões e estilos devem ser estabelecidos no início e seguidos por todos

Princípio #4

- O que é produzido por você é consumido por outros
  - Sempre especifique, projete e codifique algo pensando que **outros vão ler**
  - Sempre exija qualidade nos produtos que você consome e forneça qualidade nos produtos que você produz

Princípio #5

- Esteja pronto para o futuro
  - Sistemas de boa qualidade têm **vida longa**
  - Projete desde o início **pensando na manutenção**

Princípio #6

- Planeje para reutilização
  - Pense no **problema geral**, e não só no problema específico
  - Busque por soluções já existentes

Princípio #7

- Pense!
  - “Plano é desnecessário, mas planejar é indispensável” – D. Eisenhower
  - Avalie alternativas
  - Mitigue os riscos

Quiz

1. A Engenharia de Software é aplicação de uma abordagem sistemática, disciplinada e quantificativa para o desenvolvimento, operação e manutenção de software.

VERDADEIRO

FALSO

Quiz

2. Quais são os elementos básicos da Engenharia de Software?

a. Processos, pessoas, organização
b. Processos, métodos, ferramentas
c. Projeto, equipe, cliente
d. Projeto, software, teste

Quiz

3. A Engenharia de Software surgiu como resposta aos altos custos de hardware, com o propósito de organizar o conhecimento e propor soluções economicamente viáveis.

VERDADEIRO

FALSO

Quiz

4. As principais etapas da Engenharia são:
   a. Planejamento, execução e finalização
   b. Análise, síntese e correções
   c. Definição, medição e correções
   d. Planejamento, medição e finalização

Quiz

5. Um bom engenharia de software deve saber:
   a. Só saber programar bem
   b. Só saber planejar e executar bem o projeto
   c. Só conhecer o processo de desenvolvimento, saber medi-lo e melhorá-lo
   d. Todas as opções anteriores

Quiz

6. Basta selecionar uma excelente ferramenta de apoio ao desenvolvimento de software que a maioria dos problemas em Engenharia de Software serão resolvidos.

VERDADEIRO

FALSO

Quiz

7. “Se estivermos com o cronograma atrasado, basta adicionar mais gente ao projeto”

VERDADEIRO

FALSO

Quiz

8. Fazer software, em última instância, consiste em agregar valor para o usuário

VERDADEIRO

FALSO

Referências

- Slides Engenharia de Software – Professor Leonardo Murta
- Slides Engenharia de Software – Professor Marcelo Schots
- Pressman, R.S.; “Engenharia de Software”; 6ª edição, Ed. McGraw-Hill, 2006
- Slides Introdução à Engenharia de Softaware, Professor Márcio Barros