

# Engenharia de Software: Uma Visão Geral

---

Engenharia de Software  
Profa. Dra. Elisa Yumi Nakagawa  
1º semestre de 2016

# Software e Engenharia de Software

## TÓPICOS

- A importância do Software
- Software
- Aplicações de Software
- Mitos de Software
- Processo de Software
- Modelos de Processo de Software

# SOFTWARE

---

- INSTRUÇÕES  
que quando executadas produzem a função e o desempenho desejados
- ESTRUTURAS DE DADOS  
que possibilitam que os programas manipulem adequadamente a informação
- DOCUMENTOS  
que descrevem a operação e o uso dos programas

# Características do Software

---

- Desenvolvido ou projetado por engenharia
- Não manufacturado no sentido clássico
- Não se desgasta, mas se deteriora

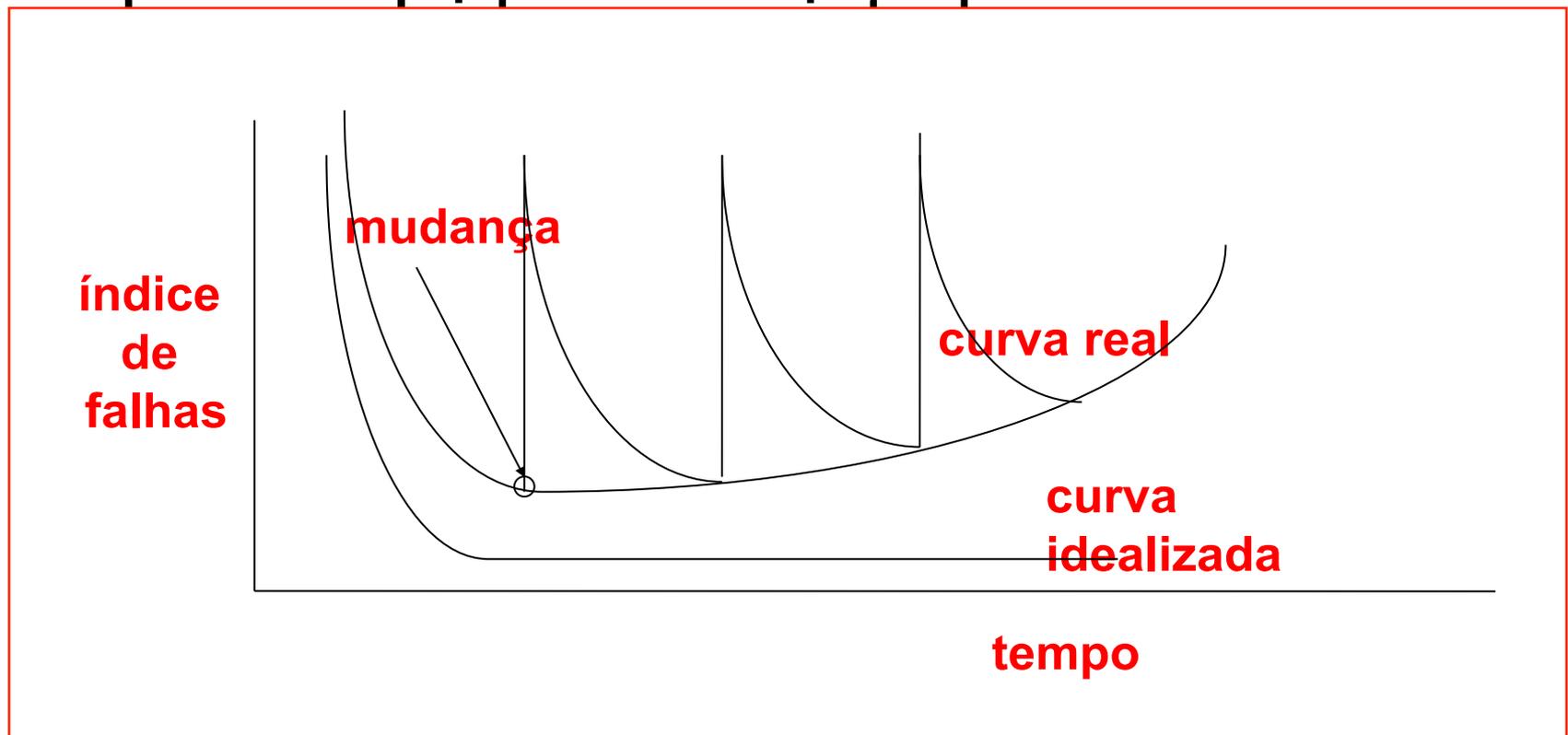
# Características do Software

● desenvolvido ou projetado por



**CURVA DE FALHAS DO HARDWARE**

# Características do Software



**CURVA DE FALHAS DO SOFTWARE**

# Características do Software

- Desenvolvido ou projetado por engenharia
- Não manufaturado no sentido clássico
- Não se desgasta mas se deteriora
- A maioria é feita ainda sob medida, em vez de ser montada a partir de componentes existentes

# Aplicações do Software

- BÁSICO
- DE TEMPO REAL
- COMERCIAL
- CIENTÍFICO E DE ENGENHARIA
- EMBUTIDO
- DE COMPUTADOR PESSOAL
- DE INTELIGÊNCIA ARTIFICIAL
- .....

# Aplicações do Software

- **BÁSICO** coleção de programas escritos para dar apoio a outros programas
- **DE TEMPO REAL** software que monitora, analisa e controla eventos do mundo real
- **COMERCIAL** sistemas de operações comerciais e tomadas de decisões administrativas

# Aplicações do Software

- CIENTÍFICO E DE ENGENHARIA caracterizado por algoritmos de processamento de números
- EMBUTIDO usado para controlar produtos e sistemas para os mercados industriais e de consumo
- DE COMPUTADOR PESSOAL envolve processamento de textos, planilhas eletrônicas, diversões, etc.

# Aplicações do Software

---

- DE INTELIGÊNCIA ARTIFICIAL faz uso de algoritmos não numéricos para resolver problemas que não sejam favoráveis à computação ou à análise direta

# Evolução do Software

(1950 - 1965)

- ⇒ O hardware sofreu contínuas mudanças
- ⇒ O software era uma arte "secundária" para a qual havia poucos métodos sistemáticos
- ⇒ O hardware era de propósito geral
- ⇒ O software era específico para cada aplicação
- ⇒ Não havia documentação

# Evolução do Software

(1965 - 1975)

- ⇒ Multiprogramação e sistemas multiusuários
- ⇒ Sistemas de tempo real
- ⇒ 1ª geração de SGBD's
- ⇒ Produto de software - *software houses*
- ⇒ Bibliotecas de software

# Evolução do Software

(1965 - 1975)

- ⇒ Cresce o número de sistemas baseado em computador
- ⇒ Manutenção quase impossível

**..... *CRISE DE SOFTWARE***

# Evolução do Software

---

(1975 - *hoje*)

- ⇒ Sistemas distribuídos
- ⇒ Redes locais e globais
- ⇒ Uso generalizado de microprocessadores  
- produtos inteligentes
- ⇒ Hardware de baixo custo
- ⇒ Impacto de consumo

# Evolução do Software

## Atualmente

- ⇒ Tecnologias orientadas o objetos
- ⇒ Sistemas especialistas e software de inteligência artificial usados na prática
- ⇒ Software de rede neural artificial
- ⇒ Computação paralela
- ⇒ Várias outras tecnologias

# Evolução do Software

(1965 - 1970)

*AFLIÇÃO CRÔNICA*

⇒ Cresce o número de sistemas baseados em computadores

⇒ Manutenção

*CRISE DE SOFTWARE*

Refere-se a um conjunto de **problemas** encontrados no desenvolvimento de software

# Crise de Software - problemas

- 1- Estimativas de prazo e de custo frequentemente são imprecisas
  - “ Não dedicamos tempo para coletar dados sobre o processo de desenvolvimento de software”
  - “Sem nenhuma indicação sólida de produtividade, não podemos avaliar com precisão a eficácia de novas ferramentas, métodos ou padrões”

# Crise de Software - problemas

---

## 2- Insatisfação do cliente com o sistema concluído

- “Os projetos de desenvolvimento de software normalmente são efetuados apenas com um vago indício das exigências do cliente”

# Crise de Software - problemas

---

- 3- Qualidade de software às vezes é menos que adequada
  - Só recentemente começam a surgir conceitos quantitativos sólidos de garantia de qualidade de software

# Crise de Software - problemas

---

4- Software existente é muito difícil de manter

- A tarefa de manutenção devora o orçamento destinado ao software
- A facilidade de manutenção não foi enfatizada como um critério importante

# Causas dos problemas associados à crise de software

## 1- PRÓPRIO CARÁTER DO SOFTWARE

O software é um elemento de sistema lógico e não físico. Conseqüentemente, o sucesso é medido pela qualidade de uma única entidade e não pela qualidade de muitas entidades manufaturadas

**Software não se desgasta, mas se deteriora**

# Causas dos problemas associados à crise de software

## 2- FALHAS DAS PESSOAS RESPONSÁVEIS PELO DESENVOLVIMENTO DE SOFTWARE

- Gerentes sem nenhum *background* em software
- Profissionais da área de software têm pouco treinamento formal em novas técnicas para o desenvolvimento de software
- Resistência às mudanças

# Causas dos problemas associados à crise de software

## 3- MITOS DO SOFTWARE

Propagaram desinformação e confusão

- ⇒ ***administrativos***
- ⇒ ***cliente***
- ⇒ ***profissional***

# Mitos do software

## ADMINISTRATIVOS:

### Mito 1:

- Já temos um manual repleto de padrões e procedimentos para a construção de software.
- Isso não oferecerá ao meu pessoal tudo o que eles precisam saber?

# Mitos do software

## ADMINISTRATIVOS:

### Realidade:

- *Será que o manual é usado?*
- *Os profissionais sabem que ele existe?*
- *Ele reflete a prática moderna de desenvolvimento de software?*
- *Ele é completo?*

# Mitos do software

---

## ADMINISTRATIVOS:

### Mito 2:

- Meu pessoal tem ferramentas de desenvolvimento de software de última geração.

# Mitos do software

## ADMINISTRATIVOS:

### Realidade:

- *É preciso muito mais do que os mais recentes computadores e ferramentas para se fazer um desenvolvimento de software de alta qualidade.*

# Mitos do software

---

## ADMINISTRATIVOS:

### Mito 3:

- Se nós estamos atrasados nos prazos, podemos adicionar mais programadores e tirar o atraso

# Mitos do software

## ADMINISTRATIVOS:

### Realidade:

- *O desenvolvimento de software não é um processo mecânico igual à manufatura. Acrescentar pessoas em um projeto torna-o ainda mais atrasado.*
- *Pessoas podem ser acrescentadas, mas somente de uma forma planejada.*

# Mitos do software

---

## CLIENTE:

### Mito 1:

- Uma declaração geral dos objetivos é suficiente para se começar a escrever programas. Podemos preencher os detalhes mais tarde.

# Mitos do software

## CLIENTE:

### Realidade:

- *Uma definição inicial ruim é a principal causa de fracassos dos esforços de desenvolvimento de software.*
- *É fundamental uma descrição formal e detalhada do domínio da informação, função, desempenho, interfaces, restrições de projeto e critérios de validação.*

# Mitos do software

---

## CLIENTE:

### Mito 2:

- Os requisitos de projeto modificam-se continuamente, mas as mudanças podem ser facilmente acomodadas, porque o software é flexível.

# Mitos do software

## CLIENTE:

### Realidade:

- *Uma mudança, quando solicitada tardiamente num projeto, pode ser maior do que a ordem de magnitude mais dispendiosa da mesma mudança solicitada nas fases iniciais.*

# Mitos do software

## MAGNITUDE DAS MUDANÇAS

<b>FASES</b>	<b>CUSTO DE MANUTENÇÃO</b>
<b>DEFINIÇÃO</b>	<b>1 x</b>
<b>DESENVOLVIMENTO</b>	<b>1.5 - 6x</b>
<b>MANUTENÇÃO</b>	<b>60 - 100x</b>

# Mitos do software

---

## PROFISSIONAL:

### Mito 1:

- Assim que escrevermos o programa e o colocarmos em funcionamento, nosso trabalho estará completo.

# Mitos do software

## PROFISSIONAL:

### Realidade:

- *Os dados da indústria indicam que entre 50 e 70% de todo esforço gasto num programa serão despendidos depois que ele for entregue pela primeira vez ao cliente.*

# Mitos do software

## PROFISSIONAL:

### Mito 2:

- Enquanto não tiver o programa "funcionando", eu não terei realmente nenhuma maneira de avaliar sua qualidade.

# Mitos do software

## PROFISSIONAL:

### Realidade:

- *Um programa funcionando é somente uma parte de uma Configuração de Software que inclui todos os itens de informação produzidos durante a construção e manutenção do software.*

# Resposta à Crise de Software

## Engenharia de Software

A aplicação de uma abordagem sistemática, disciplinada e possível de ser medida para o desenvolvimento, operação e manutenção do software  
(*IEEE*)

# Resposta à Crise de Software

## PROCESSO DE SOFTWARE

A aplicação de uma **abordagem** sistemática, disciplinada e possível de ser medida para o desenvolvimento, operação e manutenção do software (*IEEE*)

# Processo de Software

---

- Abrange um conjunto de três elementos fundamentais:

**Métodos, Ferramentas e Procedimentos**  
para projetar, construir e manter grandes sistemas de software de forma profissional

# Processo de Software

- **MÉTODOS:** proporcionam os detalhes de como fazer para construir o software
  - 👉 **Planejamento e estimativa de projeto**
  - 👉 **Análise de requisitos de software e de sistemas**
  - 👉 **Projeto da estrutura de dados**
  - 👉 **Algoritmo de processamento**
  - 👉 **Codificação**
  - 👉 **Teste**
  - 👉 **Manutenção**

# Processo de Software

- **FERRAMENTAS:** dão suporte automatizado aos métodos.
  - Existem atualmente ferramentas para sustentar cada um dos métodos
  - Quando as ferramentas são integradas, é estabelecido um sistema de suporte ao desenvolvimento de software chamado *CASE - Computer Aided Software Engineering*

# Processo de Software

- **PROCEDIMENTOS:** constituem o elo de ligação entre os métodos e ferramentas
  - Sequência em que os métodos serão aplicados
  - Produtos que se exige que sejam entregues
  - Controles que ajudam assegurar a qualidade e coordenar as alterações
  - Marcos de referência que possibilitam administrar o progresso do software.

# Um Processo de Software com Qualidade

- A *Qualidade do Processo de Software* está relacionada à extensão na qual um processo de software específico é *eficiente* e é explicitamente *definido, gerenciado, medido* e *controlado*.
- A Qualidade de Processo de Software também implica em um *potencial* para crescimento na capacidade do processo de software e a *consistência* com a qual ele é aplicado em projetos por toda a organização.

# Um Processo de Software com Qualidade (SOMMERVILLE)

- Inteligibilidade
  - o processo é definido e inteligível
- Visibilidade
  - o progresso do processo é visível externamente
- Suportabilidade
  - o processo pode ser apoiado por ferramentas CASE

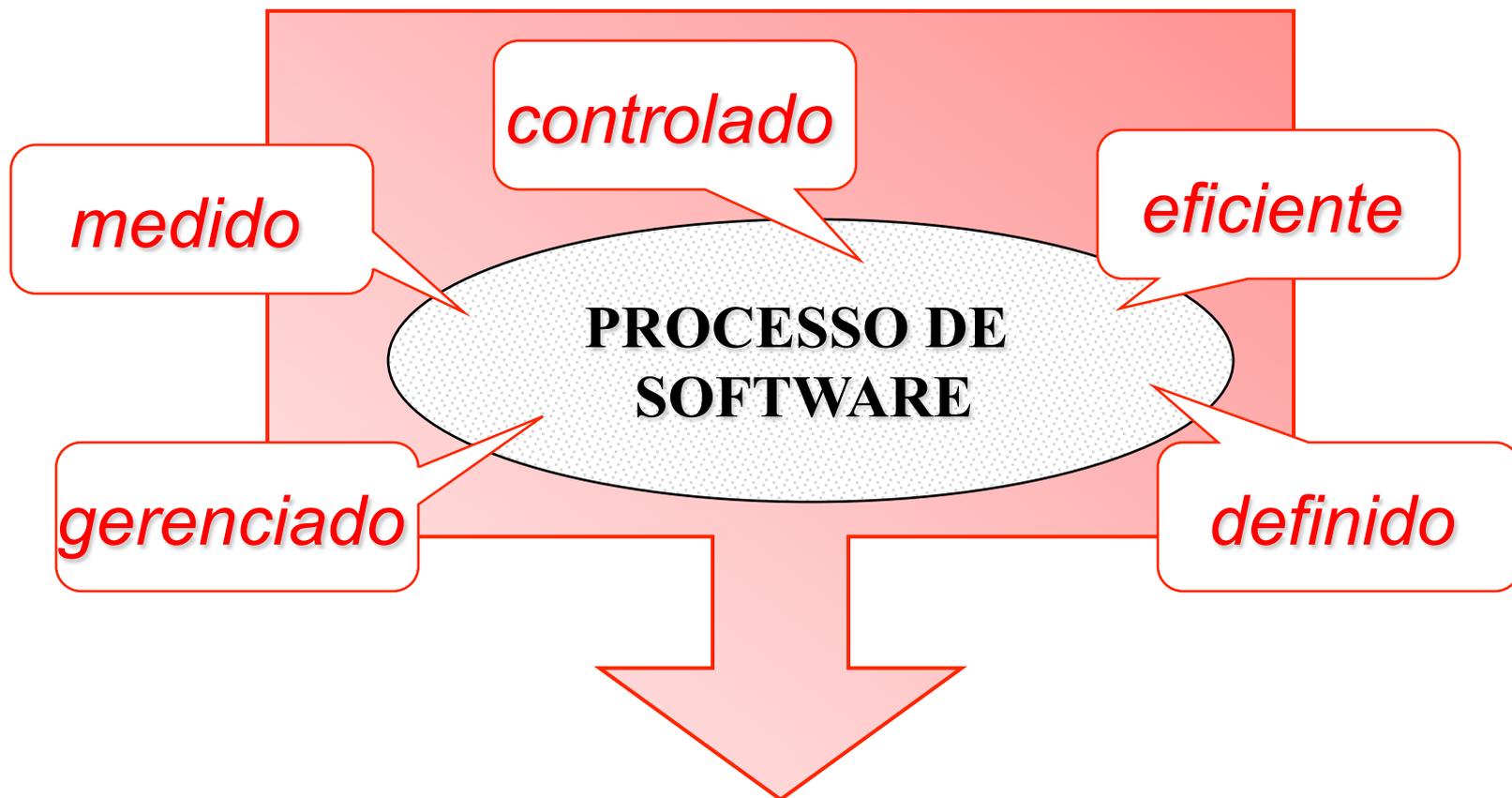
# Um Processo de Software com Qualidade (SOMMERVILLE)

- Aceitabilidade
  - o processo é aceito por todos envolvidos nele
- Confiabilidade
  - os erros do processo são descobertos antes que resultem em erros no produto
- Robustez
  - o processo pode continuar a despeito de problemas inesperados

# Um Processo de Software com Qualidade (SOMMERVILLE)

- Manutenibilidade
  - o processo pode evoluir para atender alterações de necessidades organizacionais
- Velocidade
  - quanto rápido o sistema pode ser produzido

# Um Processo de Software com Qualidade



**MODELOS DE PROCESSO DE SOFTWARE**

# Fases Genéricas dos Modelos de Processo de ENGENHARIA

- Especificação - estabelecer os requisitos e restrições do sistema
- Projeto - produzir um modelo documentado do sistema
- Implementação - construir o sistema
- Teste - verificar se o sistema atende às especificações requeridas
- Instalação - liberar o sistema para o cliente e garantir que ele seja operacional
- Manutenção – eliminar defeitos e evoluir o sistema conforme demanda.

# Fases Genéricas dos Modelos de Processo de SOFTWARE

- Independentemente da natureza do projeto e aplicação os modelos de processo de software possuem:
  - fase de definição
  - fase de desenvolvimento
  - fase de manutenção
  - atividades de apoio

# Fase de Definição do Processo de Software

focaliza "*o que*" será desenvolvido

- que informação vai ser processada
- que função e desempenho são desejados
- que comportamento pode ser esperado do sistema
- que interfaces vão ser estabelecidas
- que restrições de projeto existem
- que critérios de validação são exigidos para definir um sistema bem sucedido
- que tarefas serão realizadas

# Fase de Definição do Processo de Software

focaliza "o que" será desenvolvido

*três tarefas principais ocorrem de alguma forma:*

*engenharia de sistemas*

*planejamento do projeto de software*

*análise de requisitos*

- que critérios de validação são exigidos para definir um sistema bem sucedido

# Fase de Desenvolvimento do Processo de Software

Focaliza "como" o software será desenvolvido

- como os dados vão ser estruturados
- como a função vai ser implementada como uma arquitetura de software
- como os detalhes procedimentais vão ser implementados
- como as interfaces vão ser caracterizadas
- como o projeto será traduzido em uma linguagem de programação
- como os testes serão efetuados

# Fase de Desenvolvimento do Processo de Software

- Focaliza "como" o software será desenvolvido

*três tarefas técnicas específicas deverão ocorrer sempre:*

*projeto de software*

*geração de código*

*Inspeção e teste de software*

- como o projeto será traduzido em uma linguagem de programação
- como os testes serão efetuados

# Fase de Manutenção do Processo de Software

focaliza as "*mudanças*" que ocorrerão depois que o software for liberado para uso operacional

- A fase de manutenção reaplica os passos das fases de definição e desenvolvimento, mas faz isso no contexto de um software existente.

# Fase de Manutenção do Processo de Software

- focaliza as "*mudanças*" que ocorrerão depois

*As mudanças estão associadas com*

- *correção* de erros/defeitos
- *adaptações* exigidas conforme o ambiente do software evolui
- *mudanças* devido a *melhoramentos* ocorridos por alterações nos requisitos dos clientes

# Atividades de Apoio ao Processo de Software

- As três fases genéricas do processo de software são complementadas por uma série de *atividades de apoio*.
- As atividades de apoio são aplicadas durante toda a engenharia do software

# Atividades de Apoio ao Processo de Software

*Atividades típicas nessa categoria são:*

- *Controle e Acompanhamento do Projeto de Software*
- *Revisões Técnicas Formais*
- *Garantia de Qualidade de Software*
- *Gerenciamento de Configuração de Software*
- *Preparação e Produção de Documentos*
- *Gerenciamento de Reusabilidade*
- *Medidas*

# Modelos de Processo de Software

- Existem vários *modelos de processo de desenvolvimento de software* (ou *paradigmas de engenharia de software*)
- Cada um representa uma tentativa de colocar ordem em uma atividade inerentemente caótica
- Pode-se citar os seguintes *modelos de processo de desenvolvimento de software*

# Modelos de Processo

- *O Modelo Sequencial Linear*
  - *(também chamado Ciclo de Vida Clássico ou Modelo Cascata)*
- *O Paradigma de Prototipação*
- *Técnicas de Quarta Geração*
- *O Modelo RAD (Rapid Application Development)*
- *Modelos de Métodos Formais*
- *Modelos Evolutivos de Processo de Software*
  - *O Modelo Incremental*
  - *O Modelo Espiral*
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*