

# *Manutenção de Software*

---

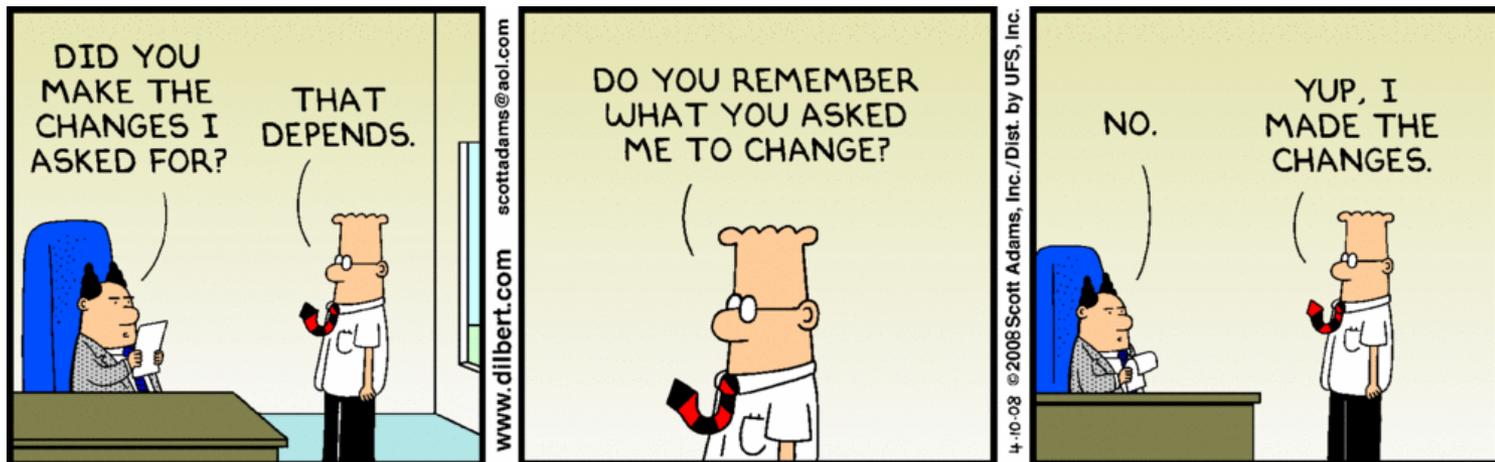
Engenharia de Software

Rosana Braga

(material produzidos por docentes do Labes-ICMC/USP)

# Manutenção do software

- O propósito do processo manutenção do sistema e software é modificar o produto de sistema/software depois de liberado.





# *Manutenção de Software*

---

- As alterações ocorrem por diversas razões.
- As razões para as alterações determinam a categoria de manutenção.



# ***Categorias de Manutenção***

---

1) Identificar e Corrigir Erros

⇒ *Manutenção Corretiva*



# ***Categorias de Manutenção***

---

2) Adaptar o Software ao Ambiente

⇒ *Manutenção Adaptativa*



## ***Categorias de Manutenção***

---

3) Atender Pedidos do Usuário para Modificar Funções Existentes, Incluir Novas Funções e Efetuar Melhoramentos Gerais

⇒ **Manutenção Perfectiva**



## ***Categorias de Manutenção***

---

4) Melhorar a manutenibilidade ou confiabilidade futuras e fornecer uma base melhor para futuras melhorias

⇒ **Manutenção Preventiva**



# *A Fase de Manutenção de Software*

---

- Fase mais problemática do Ciclo de Vida de Software
- Pode consumir mais de 70% de todo esforço do desenvolvimento
- Os sistemas devem continuar executando e as alterações são inevitáveis

# *A Fase de Manutenção de Software*



---

- Por que é exigida tanta Manutenção e por que é consumido tanto Esforço nessa atividade?
  - Idade média de 10 a 15 anos
  - Principal Interesse na época: Tamanho do Programa e Espaço de Armazenamento
  - Migração Para Novas Plataformas
  - Sistemas mal estruturados

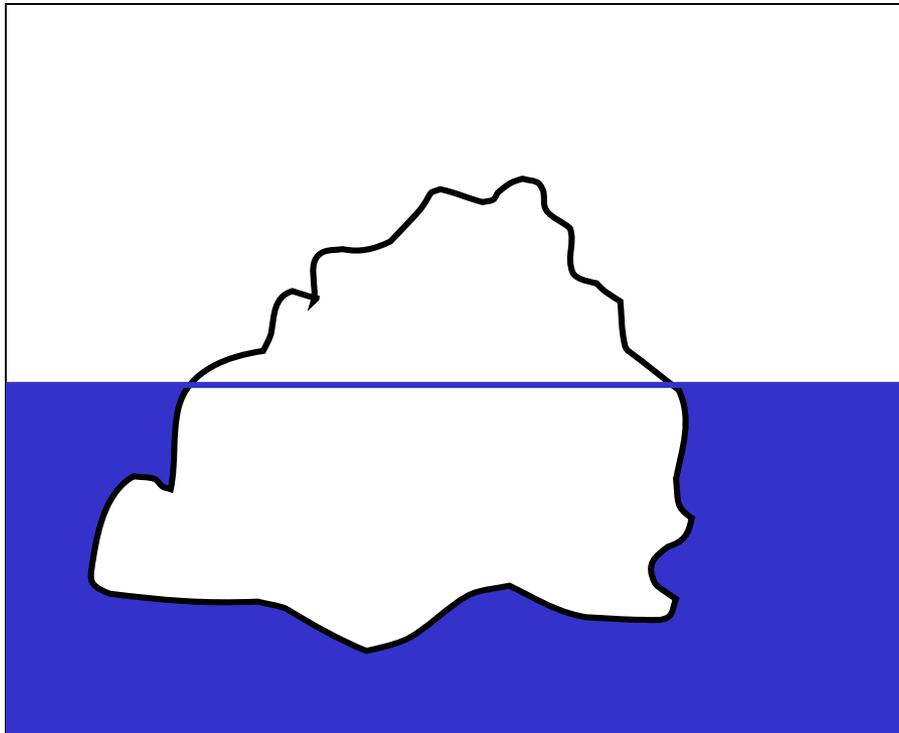
# A Fase de Manutenção de Software



---

- Por que é exigida tanta Manutenção e por que é consumido tanto Esforço nessa atividade?
  - Melhoramentos para atender novas necessidades
  - Nenhuma preocupação com a Arquitetura Global
  - Codificação, Lógica e Documentação ruins

# *Manutenção de Software*



**"iceberg da  
manutenção"**

muitos  
problemas e  
custos não são  
visíveis ou  
conhecidos



# *Custo de Manutenção*

---

- O Custo de manutenção pode ser dividido em:
  - tentar entender o que o software faz
  - interpretar as estruturas de dados, as características de interface e limites de desempenho
  - analisar, avaliar, projetar, codificar e testar as modificações

*(Atividades Produtivas)*



# *Custo de Manutenção*

---

## MODELO DE ESFORÇO DE MANUTENÇÃO

$$M = p + ke^{(c-d)}$$

onde

- $M$  = Esforço total
- $p$  = Esforço produtivo
- $k$  = Constante empírica
- $c$  = Medida de complexidade atribuída à falta de bom projeto e documentação
- $d$  = Medida do grau de familiaridade com o software



# ***Problemas da Manutenção***

---

## **PROBLEMAS CLÁSSICOS**

- É difícil ou impossível traçar a evolução do software através das várias versões. As alterações não são adequadamente documentadas
- É difícil ou impossível traçar o processo através do qual o software foi criado.



# ***Problemas da Manutenção***

---

## **PROBLEMAS CLÁSSICOS**

- É muito difícil entender programas "de outras pessoas". A dificuldade aumenta conforme o número de elementos na configuração de software diminui.
- "As outras pessoas" freqüentemente não estão presentes para explicar.



# ***Problemas da Manutenção***

---

## **PROBLEMAS CLÁSSICOS**

- A documentação não existe, é incompreensível ou está desatualizada.
- A maioria dos softwares não foram projetados para suportar alterações.
- A manutenção não é vista como um trabalho glamoroso



# ***Manutenibilidade***

---

- A Manutenibilidade pode ser definida qualitativamente como a facilidade com que o software pode ser entendido, corrigido, adaptado e ou melhorado



# ***Manutenibilidade***

---

- A manutenibilidade é afetada por muitos fatores:
  - *cuidado inadequado com o projeto, codificação e teste*
  - *configuração de software ruim*
  - *disponibilidade de pessoal qualificado de software*



# ***Manutenibilidade***

---

- A manutenibilidade é afetada por muitos fatores:
  - *facilidade de manusear o sistema*
  - *uso de linguagens de programação padronizadas*
  - *uso de sistema operacional conhecido*
  - *estruturas padronizadas de documentação*



# *Manutenibilidade*

---

- A manutenibilidade é afetada por muitos fatores:
  - *disponibilidade da pessoa ou grupo que desenvolveu o software*
  - *o planejamento para manutenibilidade*



*(fator mais importante que afeta a manutenibilidade)*

# *Manutenibilidade: Medidas*

## *Quantitativas*



---

- Difícil de quantificar
- Pode-se determinar a manutenibilidade indiretamente considerando-se atributos das atividades de manutenção que podem ser medidos

# ***Manutenibilidade: Medidas Quantitativas***



---

## **Métricas de Manutenibilidade (Gilb)**

- *tempo de reconhecimento do problema*
- *tempo de demora administrativa (aprovar)*
- *tempo de coleta de ferramentas de manutenção*
- *tempo de análise do problema*
- *tempo de especificação da alteração*
- *tempo de correção ou modificação*
- *tempo de teste local e global*
- *tempo de revisão da manutenção*

# *Manutenibilidade: Medidas Quantitativas*



---

- A manutenibilidade pode ser medida indiretamente considerando medidas da estrutura do projeto e medidas da complexidade do software



# ***Revisões de Manutenibilidade***

---

- A manutenibilidade deve ser considerada em cada nível do *processo de revisão* da engenharia de software
- Pensar na manutenção futura durante o desenvolvimento presente!



# *Revisões de Manutenibilidade*

---

## **ETAPA DE REVISÃO DE REQUISITOS**

observar:

- áreas de melhoramentos futuros
- aspectos de portabilidade do software
- interfaces que poderiam impactar a manutenção



# *Revisões de Manutenibilidade*

---

## **ETAPA DE REVISÃO DE PROJETO**

avaliar:

- projeto arquitetural
- projeto procedimental
- projeto de interfaces e
- projeto de dados

quanto à facilidade de manutenção e a qualidade global



# *Revisões de Manutenibilidade*

---

## **ETAPA DE REVISÃO DE CÓDIGO**

dar ênfase:

- ao estilo
- à documentação interna



# *Revisões de Manutenibilidade*

---

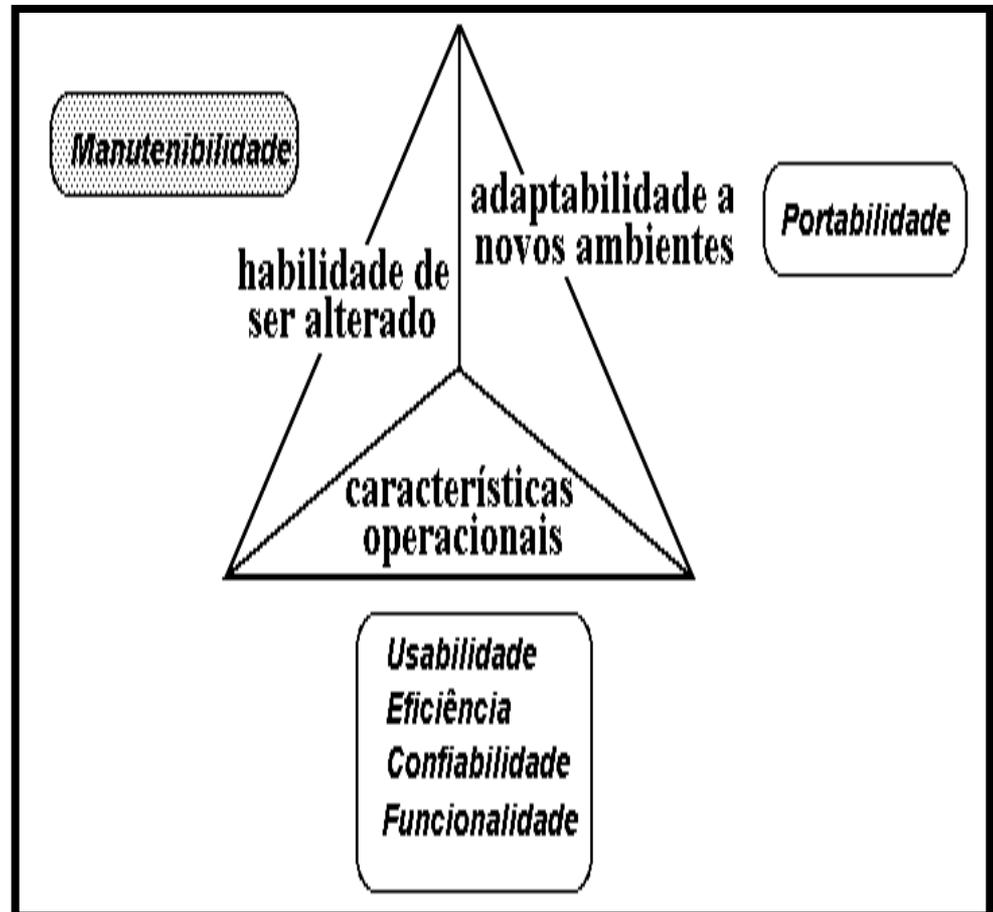
## **ETAPA DE REVISÃO DE TESTE**

- Cada passo do teste pode fornecer indícios sobre partes do software que poderiam exigir manutenção preventiva

# Fatores de Qualidade de Software

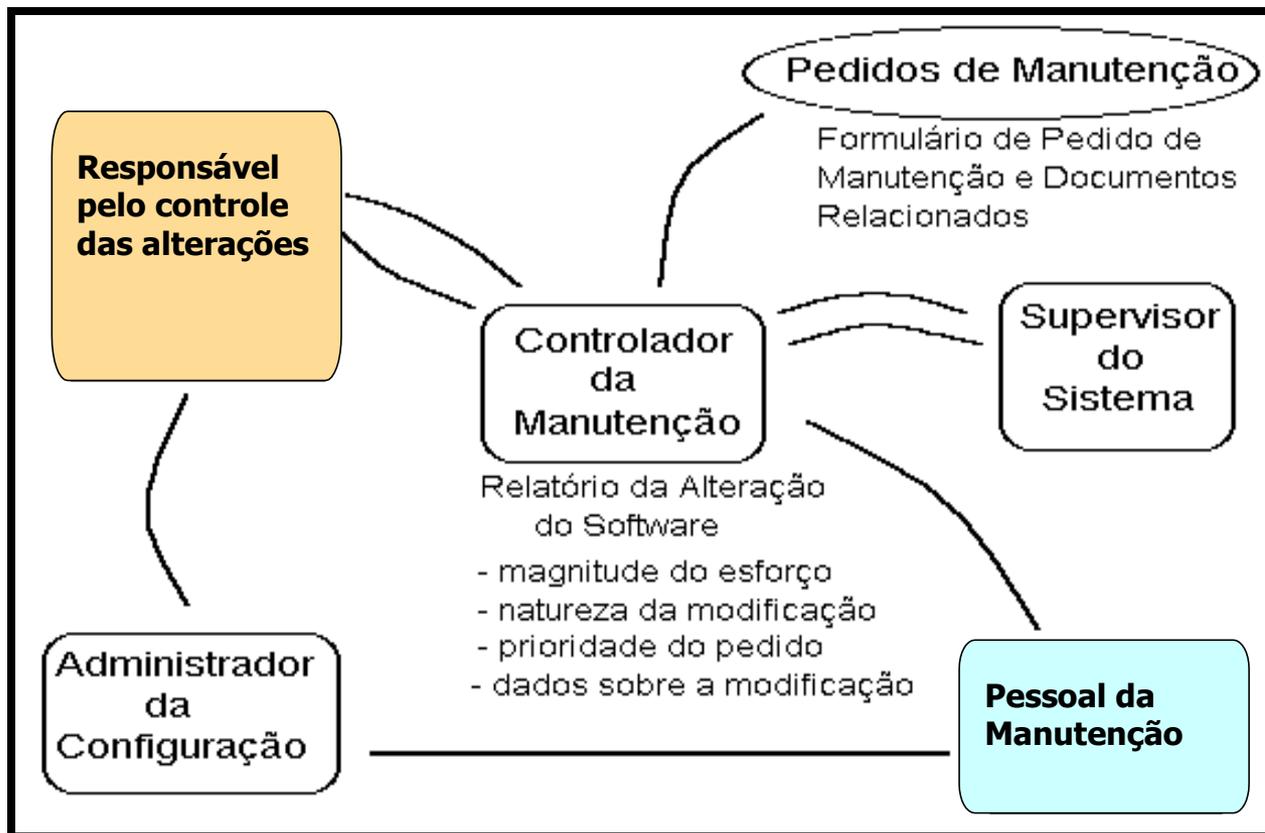
(Norma ISO 9126)

- Os fatores de qualidade de software focalizam importantes do software



# Tarefas de Manutenção

1) Estabelecer uma organização para a manutenção ("de fato" ou formal)

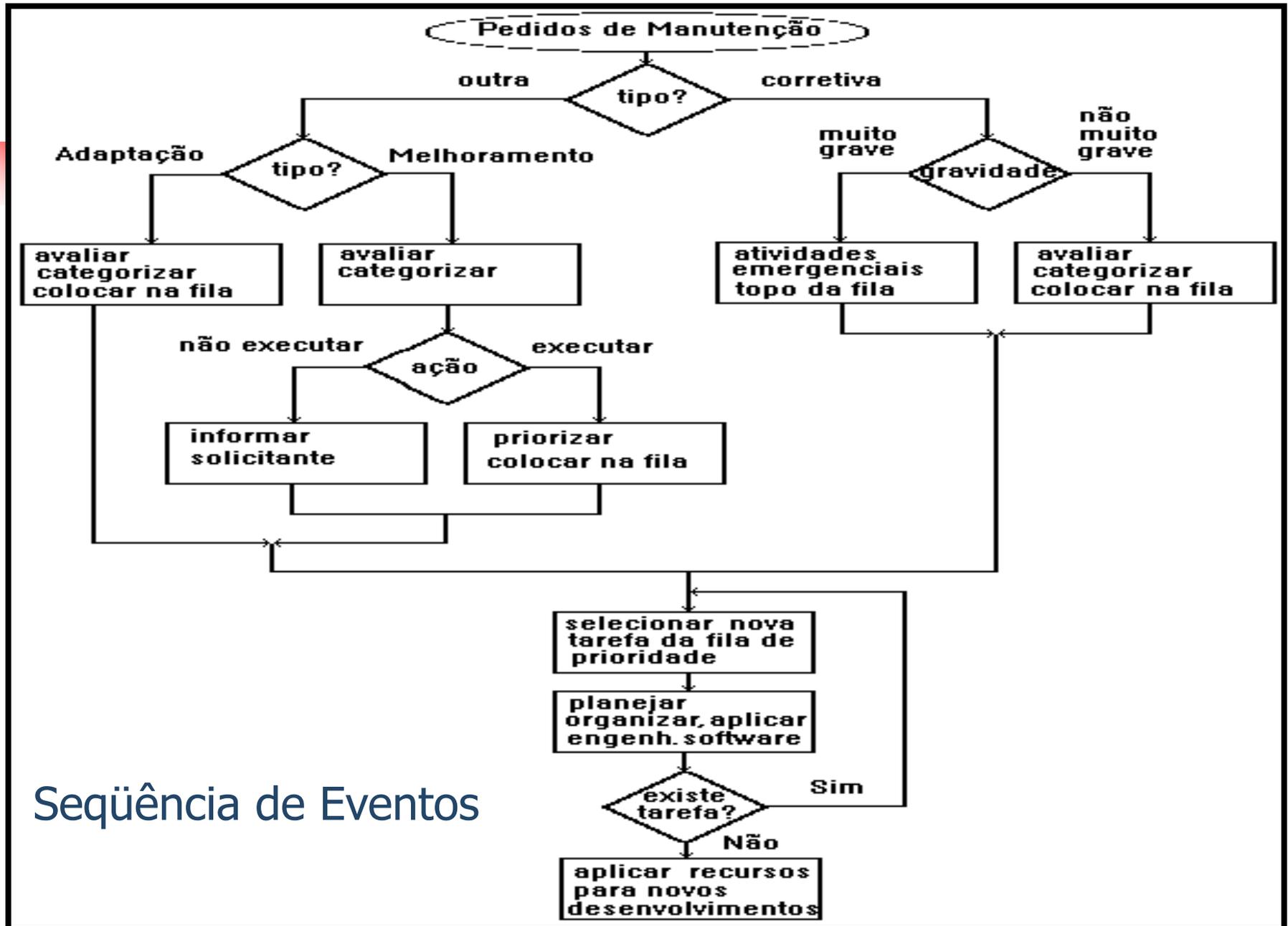




## ***Tarefas de Manutenção***

---

- 2) Descrever procedimentos de avaliação e de comunicação
- 3) Definir seqüências padronizadas de eventos (para os pedidos de manutenção)



Seqüência de Eventos



# *Tarefas de Manutenção*

---

- 4) Estabelecer procedimentos para registrar a história das atividades de manutenção

## **Exemplo de Dados que podem ser Armazenados (Swanson)**

- identificação do programa
- número de comandos fonte
- linguagem de programação usada
- data da instalação do programa
- número de execuções do programa desde a instalação
- número de falhas de processamento associadas ao item anterior
- nível e identificação da alteração no programa
- número de comandos fonte adicionados por alteração no programa
- número de pessoas-horas despendidos na manutenção
- identificação do pedido de manutenção
- tipo de manutenção
- datas de início e fim da manutenção



# ***Tarefas de Manutenção***

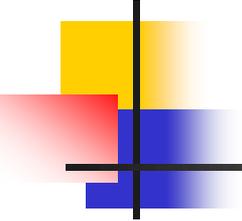
---

5) Definir critérios de revisão e avaliação

# **MEDIDAS de DESEMPENHO e MANUTENÇÃO**

(Swanson)

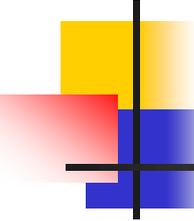
- número médio de falhas de processamento por execução do programa
- pessoas-horas consumidos em cada categoria de manutenção
- número médio de pessoas-horas despendido por comando fonte adicionado ou deletado devido à manutenção
- tempo médio de processamento para um pedido de manutenção
- porcentagem de pedidos de manutenção por tipo



---

# ***Engenharia Reversa e Reengenharia***





## *Cenário usual para manutenção*

---

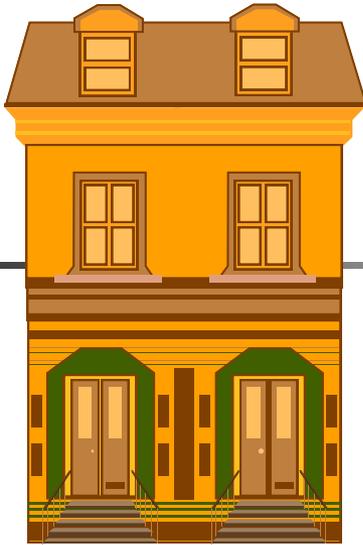
- Sistemas sem documentação
- Dificuldade de manutenção
- Erros gerando outros erros
- Código duplicado



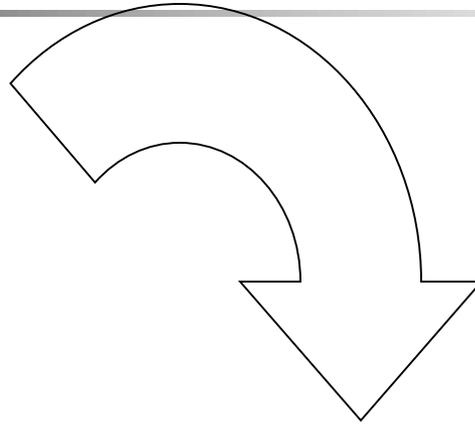
# *Necessidade de Manutenção no Software*

- O quê fazer ???
- Quem poderá me ajudar ????
- Cadê o programador ????
- O quê será que ele quis fazer aqui?????



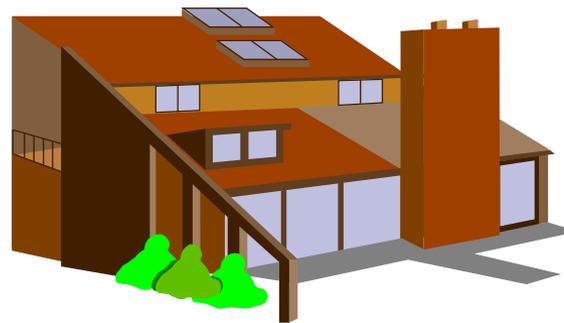


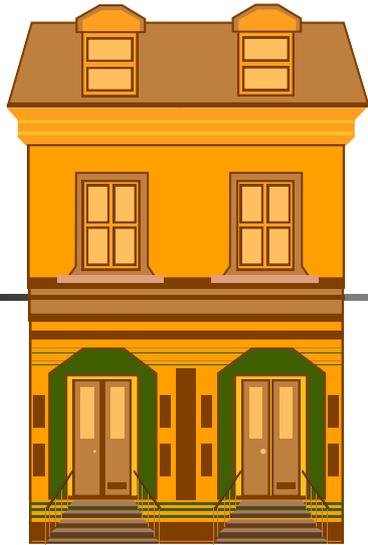
Será possível ????



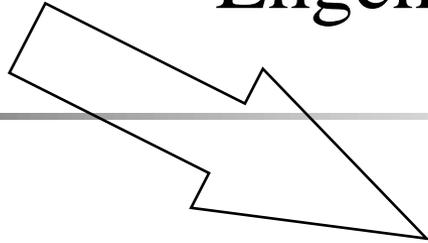
Na construção civil,  
ok

E com sistemas de  
software?

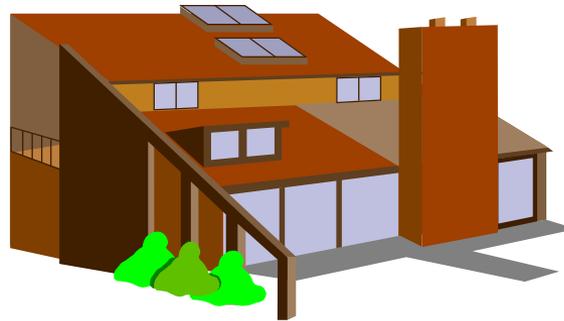
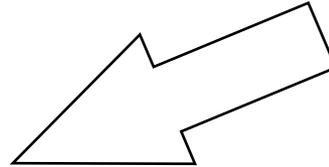
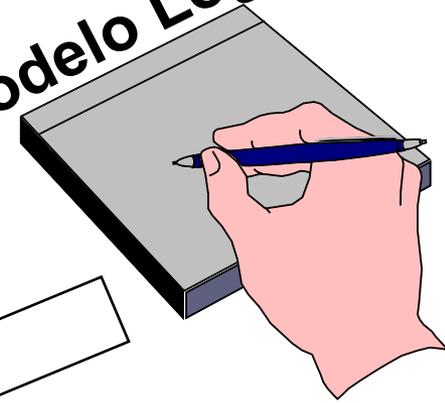




Engenharia Reversa



Modelo Lógico



Reengenharia



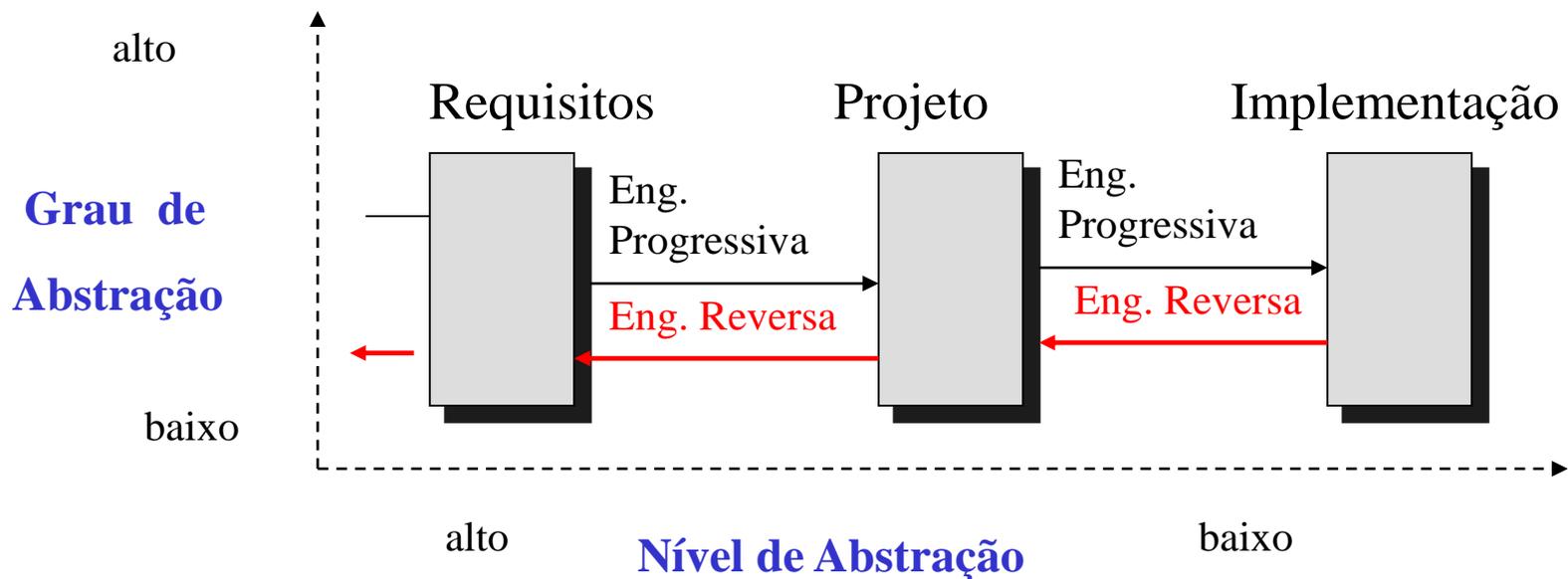
# Engenharia Reversa

---

- O termo “Engenharia Reversa” tem sua origem na análise de hardware, pois é comum a prática de decifrar projetos de produtos finalizados com intuito de duplicá-los.
- O conceito de Engenharia Reversa de Software é similar. Porém, tradicionalmente o objetivo da dessa engenharia reversa é obter apenas um entendimento do sistema

# Abstrações do Software

## ■ Engenharia Progressiva x Reversa

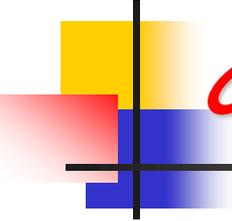


# ***Engenharia Reversa***



**Quais os documentos utilizados para realizar engenharia reversa ?**

- código fonte
- informações de usuários e/ou analista
- documentação existente: manual de usuário, manual de sistema, DFDs, fluxogramas...



## *Como começa a engenharia reversa?*

---

- obtendo-se as informações necessárias para o completo entendimento do sistema

### **O que fazer com essas informações?**

- só para manutenção
- mesmo paradigma e mudança de linguagem
- mudança de paradigma (só modelo lógico)
- mudança de paradigma e de linguagem

**Reengenharia**



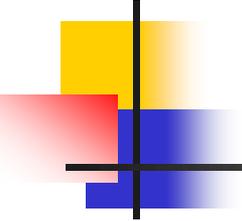
# *Reengenharia de Software é qualquer atividade que:*

---

- (1) Melhore o entendimento do software
- (2) Prepare ou melhore o software em si, aumentando sua manutenção, seu reuso e sua extensão

Chikofsky e Cross definem reengenharia:

“o exame e a alteração de um sistema para reconstituí-lo de uma nova forma, seguida pela sua implementação”



# Questões Econômicas de Engenharia Reversa

---

- Aplicar ER infringe a lei de propriedade intelectual?
  - ER do próprio software é correta e legal
- Quando o software envolvido é de propriedade alheia?
  - 1) direito exclusivo do proprietário
  - 2) cópia para estudo é legal