PUC CAMPINAS

Engenharia de Software

Tema da Aula

Manutenção de Software

Prof. Cristiano R R Portella
portella@widesoft.com.br

PUC CAMPINAS
Engenharia de Software

Manutenção de Software

- ✓ Durante as fases de desenvolvimento, defeitos podem ser inadvertidamente gerados. A revisão (testes) pode falhar ao tentar descobrir os novos erros bem como os erros provenientes de fases anteriores.
- ✓ Erros oriundos de fases anteriores, normalmente têm seus efeitos ampliados à medida em que a construção do produto de software avança (por exemplo, uma falha na estrutura de dados, irá se replicar pelo código de todos os programas que tratem essa estrutura).

PUC CAMPINAS
Engenharia de Software

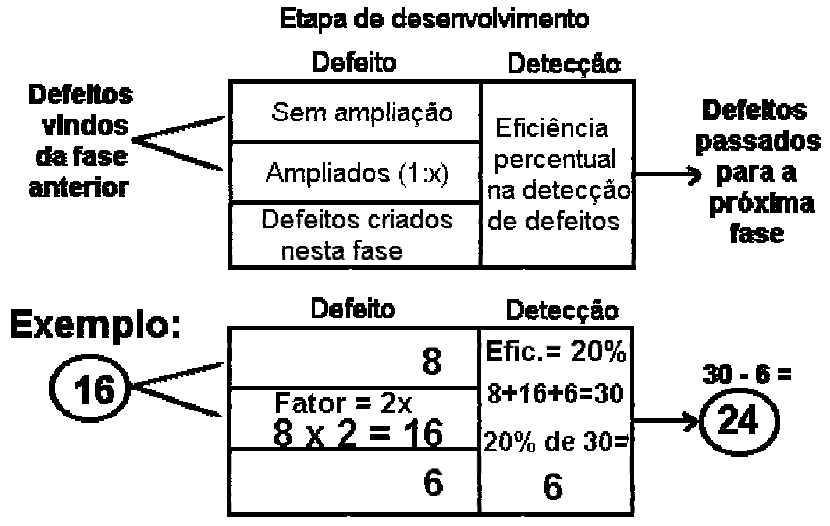
Manutenção de Software

A eficiência percentual na detecção de erros durante cada fase, aumenta na medida em que Técnicas de Testagem e padrões para Garantia da Qualidade são aplicados.

Cálculo do percentual de erros remanescentes:

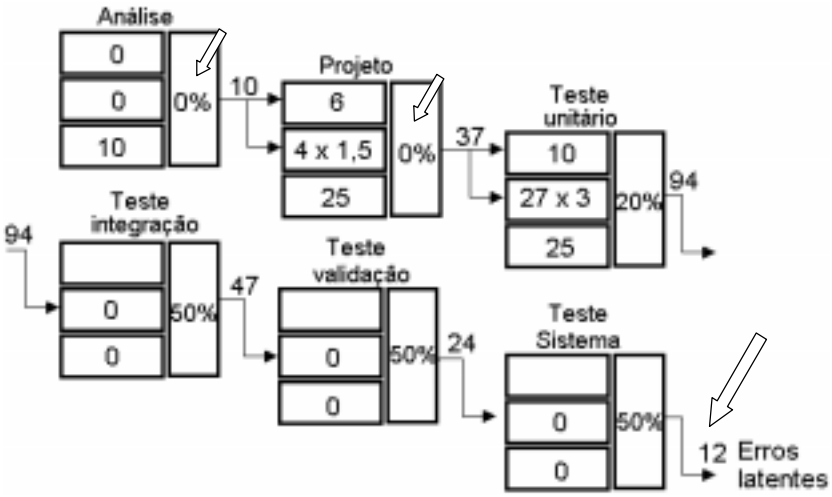
PUC CAMPINAS
Engenharia de Software

Manutenção de Software



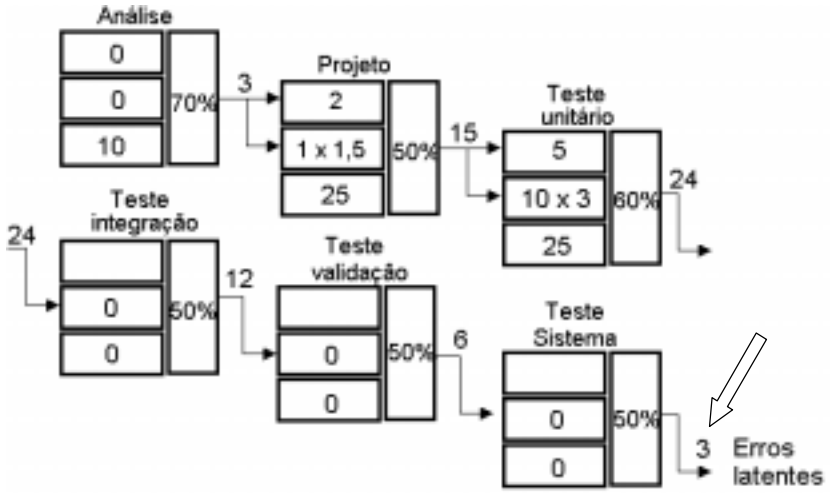
PUC CAMPINAS
Engenharia de Software

Ampliação de Defeitos
1 – Sem revisão



PUC CAMPINAS
Engenharia de Software

Ampliação de Defeitos
2 – Com Revisões Técnicas Formais



PUC CAMPINAS
Engenharia de Software

Introdução

Ainda que “reparar defeitos” seja a grande atividade da fase de Manutenção, existem outras causas que acabam por gerar a necessidade do produto de software passar por uma manutenção, tais como:.

- ✓ Mudança na legislação pertinente ao sistema;
- ✓ Mudança na tecnologia de TI/SI;
- ✓ Mudança no processo-alvo;
- ✓ Mudança no produto;
- ✓ Mudança na clientela;
- ✓ Mudança na direção;
- ✓ Mudança nos modelos de gestão etc.

PUC CAMPINAS
Engenharia de Software

Introdução

Grande parte do software do qual dependemos atualmente tem, em média, de 10 a 15 anos de idade. ..

- Criados quando o tamanho e o espaço de armazenagem eram a principal preocupação
- Portados para novas plataformas
- Ajustados às novas tecnologias de SO, HW etc
- Ampliados para atender às novas necessidades dos usuários
  - Condições em que essas manutenções se deram? (Def.Reqs, Análise, Projeto, Documentação, Testes etc etc ???)

PUC CAMPINAS
Engenharia de Software

Definição

Diferentes causas geram manutenções de tipos diferentes, que podem ser classificadas em:

- ✓ Manutenção **Corretiva**
- ✓ Manutenção **Adaptativa**
- ✓ Manutenção **Perfectiva**
- ✓ Manutenção **Preventiva**

(Engenharia Reversa e Reengenharia)

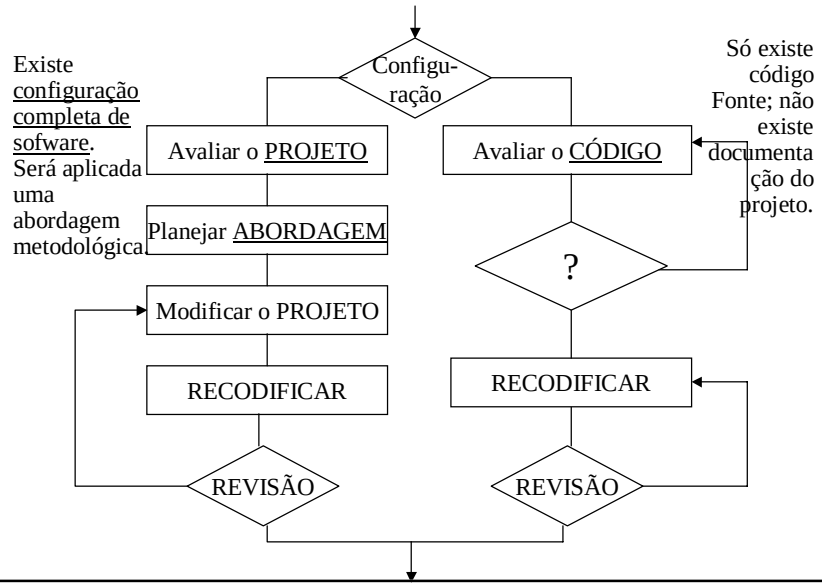
PUC CAMPINAS
Engenharia de Software

Definição

- ✓ Manutenção **Corretiva:**
  Corrigir defeitos (erros latentes)
- ✓ Manutenção **Adaptativa:**
  Adaptar-se a novas tecnologias (TI/SI), metodologias, modelos de gestão, legislação etc
- ✓ Manutenção **Perfectiva:**
  Incluir novas funções (ampliações)
- ✓ Manutenção **Preventiva:**
  Melhorar manutebilidade futura
  (Engenharia Reversa e Reengenharia)

PUC CAMPINAS
Engenharia de Software

Manutenção
Estruturada x Não-Estruturada



PUC CAMPINAS
Engenharia de Software

Configuração de Software

Configuração completa de Software:

- ✓ Estrutura de dados (Dic.Dados, Descrição das tabelas, MER, Cross-Reference etc.
- ✓ Estrutura do software (diagrama estrutural, descrição de módulos x funcionalidade, descrição de classes e s/ métodos etc.
- ✓ Interfaces do sistema (integração com outros SI's)
- ✓ Restrições de projeto
- ✓ Registro dos testes realizados
- ✓ Registro das modificações realizadas

Documentação Atualizada

PUC CAMPINAS
Engenharia de Software

Configuração de Software

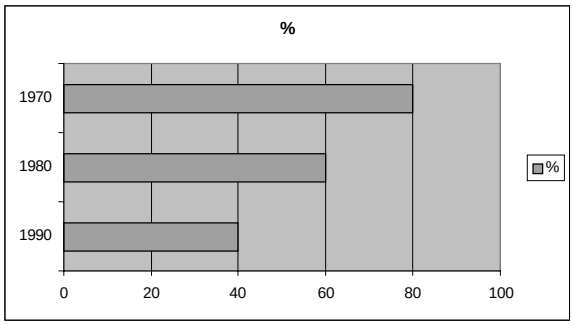
Manutenção Estruturada:

- ✓ Solicitação do usuário;
- ✓ Definir requisitos e especifica-los;
- ✓ Analisar projeto;
- ✓ Modificar projeto;
- ✓ Revisar modificações;
- ✓ Implementar código fonte;
- ✓ Testes (unitário, integração, validação e sistema);
- ✓ Registrar alterações (documentação e versão); e
- ✓ Liberar nova versão do sistema.

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Custo crescente da manutenção em relação ao custo total da área de Informática.



PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

- ✓ Custos Intangíveis:
  - Desenvolvimento postergado ou perdido.
    O maior custo da manutenção de software é uma drástica diminuição de produtividade em relação ao desenvolvimento de novos produtos.
  - Redução da qualidade global do software.
  - Insatisfação do cliente.
- ✓ Um esforço de desenvolvimento que custe 25,00 por linha de código, pode custar 1000,00 por linha mantida.

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

O esforço total de manutenção pode ser dividido em:

1. Atividades produtivas:
   Análise do projeto, avaliação da solicitação de manutenção, alteração do projeto, codificação.

2. Atividades improdutivas:
   Tentar entender o que o código faz, tentar interpretar as estruturas de dados, tentar descobrir as restrições originais do projeto, etc.

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Modelo de esforço em Manutenção:

$$Em = p + Ke^{(c-d)}$$

onde:

Em = esforço total gasto em manutenção

p = esforço produtivo

Ke = constante empírica

c = medida de complexidade atribuída à falta de um bom projeto e de uma documentação completa e atualizada

d = medida do grau de familiaridade do profissional com o projeto

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Estudo de três casos onde não ocorra variação no esforço produtivo (*p* = 10) e na constante empírica (*Ke* = 2)

1o Caso: Falta um bom projeto e uma documentação completa e atualizada (*c* = 7). O profissional tem baixa familiaridade com o software (*d* = 2).

$$Em = 10 + 2^{(7-2)}$$

$$Em = 10 + 2^{5}$$

$$Em = 42$$

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Estudo de três casos onde não ocorra variação no esforço produtivo (p = 10) e na constante empírica (Ke = 2)

2o Caso: O projeto é pior que o do primeiro caso (c = 15), porém o profissional tem familiaridade com o software (d = 10).

$$Em = 10 + 2^{(15-10)}$$

$$Em = 10 + 2^{5}$$

$$Em = 42$$

A falta de abordagem metodológica é compensada com a ancoragem do profissional de desenvolvimento, na atividade de manutenção.

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Estudo de três casos onde não ocorra variação no esforço produtivo (p = 10) e na constante empírica (Ke = 2)

3o Caso: O projeto é ruim (c = 15). Como agravante existe um alto "turnover" na equipe de desenvolvedores (rotatividade de pessoal), gerando uma baixa familiaridade com o software (d = 2).

$$Em = 10 + 2^{(15-2)}$$

$$Em = 10 + 2^{13}$$

$$Em = 8192$$

O esforço de manutenção eleva-se de maneira exponencial, pela falta de uma abordagem apoiada em princípios de Eng. Software.

PUC CAMPINAS
Engenharia de Software

Custo de Manutenção

Agravantes:

- ✓ Freqüentemente é difícil senão impossível rastrear a evolução do software através de muitas versões (falta de gerenciamento de versão).
- ✓ A codificação feita por profissionais "imaturos" profissionalmente ou sem formação ampla, não apresente legibilidade e manutebilidade.
- ✓ Geralmente as manutenções são feitas de maneira desestruturada e empírica (tentativa e erro).
- ✓ A maioria dos SW não é projetada para mudanças.
- ✓ A manutenção não é vista como um trabalho interessante.

PUC CAMPINAS
Engenharia de Software

Manutebilidade

Manutebilidade:

Facilidade com que um software pode ser entendido, corrigido, adaptado ou ampliado.

Medidas indiretas da manutebilidade de um software (Métricas de Manutenção):

- Tempo de Reconhecimento do problema
- Tempo de Análise do problema
- Tempo de Especificação das mudanças
- Tempo de Correção
- Tempo de Testes
- Tempo de Revisão da manutenção
- Tempo de Recuperação total

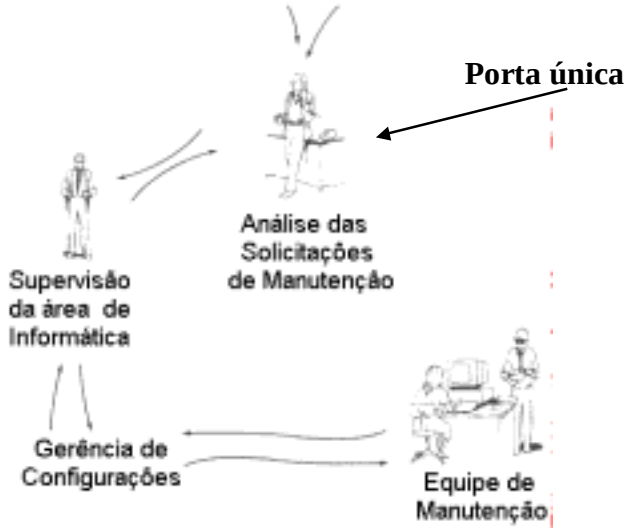
PUC CAMPINAS
Engenharia de Software

Tarefas de Manutenção

- ✓ Organização para a Manutenção
  - Porta Única
- ✓ Relatórios
  - Relatório de Problemas de SW
    (Pedido de Manutenção feito pelo usuário)
  - Relatório de Mudanças de SW
    (Relatório interno realizado a partir da solicitação do usuário. Será encaminhado para a Supervisão da área de Informática)
- ✓ Fluxo de Eventos
- ✓ Conservação de Registros
- ✓ Avaliação

PUC CAMPINAS
Engenharia de Software

Tarefas de Manutenção
Porta Única

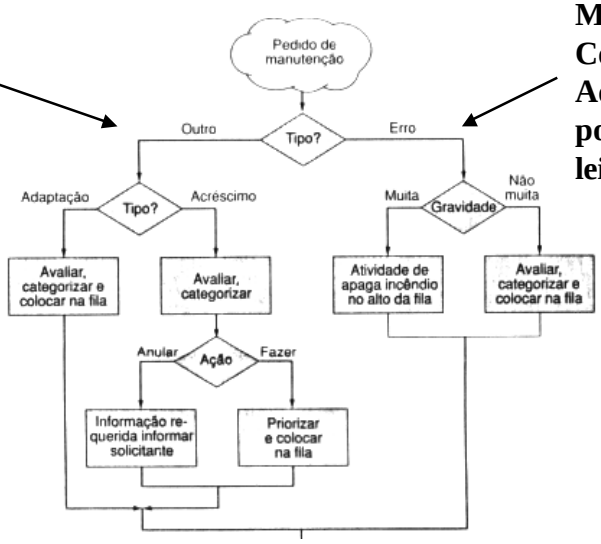


PUC CAMPINAS
Engenharia de Software

Tarefas de Manutenção
Fluxo de Eventos

Demais tipos de manutenção

Manutenção Corretiva e Adaptativa por força de lei.



PUC CAMPINAS
Engenharia de Software

Tarefas de Manutenção
Conservação de Registros

Da conservação de registros da atividade de Manutenção dependerá:

- ✓ Melhorar a precisão da estimativa de tempo para realizar futuras manutenções
- ✓ Analisar a efetividade das técnicas aplicadas em manutenção
- ✓ Realizar a atividade de Melhoria Contínua sobre os processos e controles

PUC CAMPINAS
Engenharia de Software

Tarefas de Manutenção
Conservação de Registros

Medidas em potencial, cujos dados devem ser registrados e armazenados por Sistema – programa – desenvolvedor:

- Número médio de falhas/tempo (MTBF)
- Tempo médio em manutenção (MTTM)
- Total de horas gastas por tipo de manutenção
- Sistemas com maior incidência de manutenção
- Tempo médio para atendimento de manutenção

PUC CAMPINAS
Engenharia de Software

Manutenção
Efeitos Colaterais

Qualquer mudança introduzida num procedimento lógico complexo aumenta seu potencial de erros e tende a descaracterizar o projeto original.

Assim, pode-se esperar “efeitos colaterais”

- Na Codificação
  (tratamento de arquivos, flags, operadores lógicos, testes insuficientes etc.)
- Nos Dados
  (redundâncias, perda de integridade, erro em sistemas integrados etc)
- Na Documentação
  (falta de atualização, mudança de padrão etc)

PUC CAMPINAS
Engenharia de Software

Manutenção
Códigos Alienígenas

Também conhecido como “código legado”.

✓ Características:

- Nenhum membro atual trabalhou no desenvolvimento
- Nenhuma metodologia de projeto foi aplicada
- Documentação é incompleta e registro das mudanças passadas superficial ou inexistente
- Tecnologia obsoleta

Esse tipo de manutenção pede cautela (analise e teste primeiro, não elimine o código antigo, documente tudo, faça testes exaustivos).

PUC CAMPINAS
Engenharia de Software

Formas “exóticas” de Manutenção

Ao longo de anos, correções, adaptações e expansões, aliadas às “boas práticas de ES”, levaram as aplicações à instabilidade, qualquer nova mudança causa uma série de efeitos inesperados, o tempo médio em manutenção é excessivamente alto.

Esses produtos legados têm:

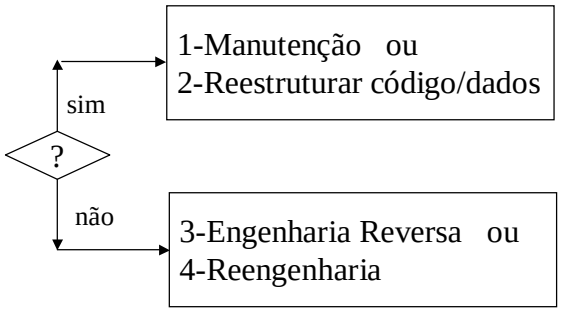
- Projeto pobre
- Codificação pobre
- Lógica pobre
- Documentação pobre

São produtos pré era da Engenharia de Software.

PUC CAMPINAS
Engenharia de Software

Formas “exóticas” de Manutenção

A questão a ser analisada é: a aplicação deve continuar a receber manutenções (reformas) ou ser reconstruída?



PUC CAMPINAS
Engenharia de Software

1 - Reestruturar Código e Dados

Reestruturar Código significa reescrever o código de alguns módulos dentro do produto de software, que estão gerando mais problemas (instáveis) ou que a manutenção exige mais esforço.

Normalmente não é escrever todo o programa, mas apenas as partes que violam regras de estruturação (por isso o termo “reestruturar” o código).

Reestruturar Dados significa alterar estruturas de dados. Geralmente as estruturas de dados impactam mais, a longo prazo, a viabilidade de um programa, que o código fonte.

PUC CAMPINAS
Engenharia de Software

1 - Reestruturar Código e Dados

Para reestruturar dados devemos analisar a estrutura atual, definir um modelo de dados e por fim refazer as estruturas normalizando-as.

Uma alteração nas estruturas de dados implica em mudanças de projeto e de codificação. Por isso, uma reestruturação deve começar pelos dados e depois passar para o código.

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

Engenharia Reversa (Forward Engineering ou Renovation ou Reclamation*) é uma forma de refazer um produto de software recuperando-se as informações de projeto existentes, as quais serão usadas para alterar o SI, num esforço de melhorar sua qualidade.

Através do exame e compreensão do software existente, num caminho oposto ao do desenvolvimento, vamos recriar o projeto e decifrar os requisitos atualmente implementados pelo sistema, apresentando-os em um nível ou grau mais alto de abstração.

(*) renovação

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

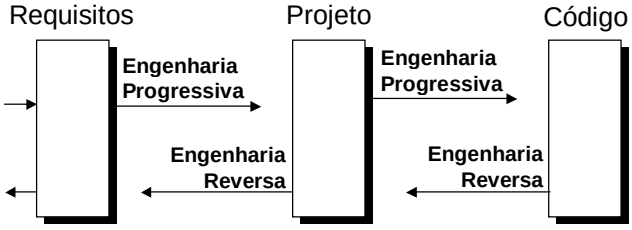
A partir do conhecimento obtido em diferentes níveis de abstração do produto (código, componentes, funcionalidade, escopo etc), um novo produto será desenvolvido, tendo as funções antigas refeitas segundo novas técnicas e tendências, além de receber novas funções, adicionadas para atender necessidades dos usuários e das novas tecnologias.

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

• **Engenharia Progressiva:** Processo tradicional de engenharia de software, caracterizado pelas atividades progressivas do ciclo de vida, que partem de um alto nível de abstração, para um baixo nível de abstração.

•**Engenharia Reversa:** O processo inverso a Engenharia Progressiva, caracterizado pelas atividades retroativas do ciclo de vida, que partem de um baixo nível de abstração para um alto nível de abstração.



PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

O conceito de Engenharia Reversa de Software é similar a sua aplicação em hardware (decifrar projetos de produtos acabados, com o intuito de duplica-los ou apenas obter um entendimento de sua arquitetura).

Definição de Engenharia Reversa: Processo de exame e compreensão do software existente, para recapturar ou recriar o projeto e decifrar os requisitos atualmente implementados pelo sistema, apresentando-os em um nível ou grau mais alto de abstração

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

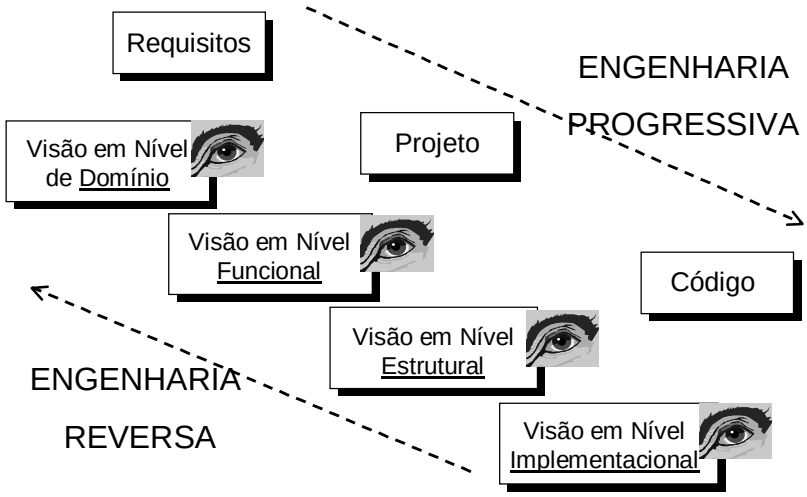
Por meio da engenharia reversa um software pode ser visualizado em diferentes níveis de abstração (código, projeto, requisitos).Cada VISUALIZAÇÃO abstrai características próprias da fase do ciclo de vida correspondente à abstração.

Baseado nos níveis de abstração, as visões são classificadas em 4 tipos:

- Visão em nível implementacional
- Visão em nível estrutural
- visão em nível funcional
- visão em nível de domínio

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa



PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

Visão em Nível Implementacional

- Abstrai características da linguagem de programação e características específicas da implementação (informações s/ sintaxe, semântica e implementação).

Visão em Nível Estrutural

- Abstrai detalhes da linguagem de programação para revelar sua estrutura a partir de diferentes perspectivas. O resultado é uma representação explícita das dependências entre os componentes do sistema (grafos tipo DFD, de Controle, projeto arquitetural etc).

2 - Engenharia Reversa

Visão em Nível Funcional

- Abstrai a função de um componente, isto é , sua característica comportamental. Essa visão relaciona partes do programa às suas funções procurando revelar as relações lógicas entre elas (diferentemente das relações sintáticas ou da estrutura), como por exemplo a relação entre processos e fluxo de dados no DFD ou a relação entre processos fluxo de controle entre eles através do Diagrama de Fluxo de Controle.

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

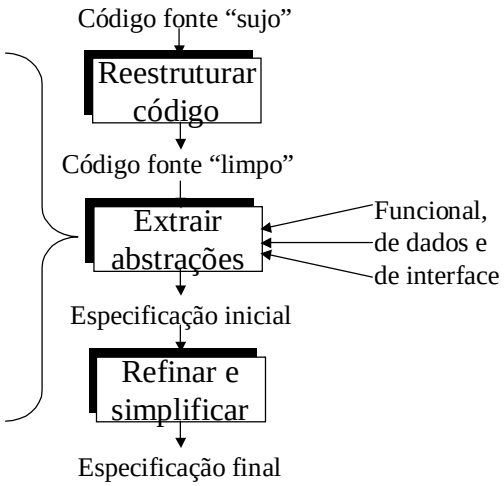
Visão em Nível de Domínio

- Abstrai o contexto em que o sistema esta operando, ou seja, as necessidades do usuário, o suporte prestado ao processo-alvo, as inter-relações entre sistemas e as restrições do contexto.

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa

Núcleo da Eng. Reversa: **Ler o código** geralmente sem documentação, em diferentes níveis de abstração: (sistema, programa, componente, padrão etc.)



PUC CAMPINAS
Engenharia de Software

Manutenção e Eng. Reversa

Manutenção (normal) e Engenharia Reversa:

Mesmo durante o desenvolvimento de alguns tipos de manutenção, aplicamos (ou pelo menos podemos aplicar) técnicas de Engenharia Reversa. Por exemplo:

Manutenção Corretiva:

Localizar componente defeituoso através da melhoria na compreensão do software.

PUC CAMPINAS
Engenharia de Software

Manutenção e Eng. Reversa

Manutenção Adaptativa:

Usar Eng.Reversa para obter visões do software a fim de localizar os componentes que receberão as mudanças/adições ou para criar a documentação necessária.

Manutenção Preventiva:

Através de visões do produto, definir onde e como realizar mudanças apropriadas. Nas futuras manutenções (preventiva), utilizar-se da documentação gerada via Eng.Reversa.

PUC CAMPINAS
Engenharia de Software

Eng. Reversa e Reuso

Reuso é uma atividade que se destina a identificar software reutilizável. Envolve também a correta importação, reconfiguração e adaptação deste software para uma nova aplicação em um sistema de computação.

O processo de reuso compreende as atividades:

- Reconhecimento
- Decomposição
- Classificação (para povoar as bibliotecas de reuso)
- Seleção
- Adaptação e Composição.

Técnicas de engenharia reversa ajudam o desenvolvimento das fases de Reconhecimento, Decomposição e Classificação.

PUC CAMPINAS
Engenharia de Software

Eng. Reversa e Reuso

Componentes candidatos a reuso podem ser mais facilmente reconhecidos, se forem convertidos para uma notação ou forma “padrão’, a exemplo da técnica denominada “Tecnologia de Grupo” utilizada na Engenharia Mecânica/Manufatura.

Mesmo que as técnicas de engenharia reversa não sejam focalizadas na identificação e composição de componentes a partir de partes reutilizáveis, ela pode ser proveitosa em completar a documentação de novos sistemas gerados a partir de componentes reutilizáveis (COTS) de várias origens (outros sistemas, bibliotecas de terceiros, Internet etc).

PUC CAMPINAS
Engenharia de Software

Eng. Reversa e Ética/Direitos Autorais

Engenharia Reversa e Questões Éticas e Legais.

Aplicar técnicas de Engenharia Reversa constitui-se numa infração da Legislação de Propriedade Industrial ou Intelectual de Software ?

A prática será legal se:

1. Temos Direito de Propriedade do Software. Não confundir com Direito de Uso.
2. No caso de Software Livre (GNU-Free Software) desde que sejam atendidas as 4 liberdades dos Termos GNU(*).
3. O desenvolvedor do software encerrou suas atividades e não existem empresas sucessoras.
4. Cópia para estudo é legal, a menos que o proprietário declare expressamente o contrário.

PUC CAMPINAS
Engenharia de Software

General Public License (GPL)

Liberdade 1:

Executar o programa com qualquer propósito;

Liberdade 2:

Estudar como o programa funciona e adapta-lo às suas necessidades. O acesso ao código fonte é um pré-requisito para que se possa estudar o software;

Liberdade 3:

Redistribuir **cópias** do programa; e

Liberdade 4:

Modificar o programa e **distribuir** suas melhorias para o público em geral, de maneira que a comunidade em geral possa se beneficiar disso

PUC CAMPINAS
Engenharia de Software

General Public License (GPL)

O autor de um software inicialmente distribuído na condição de *Software Livre*, não pode se arrepender dessa atitude e tentar revogar as liberdades. Se isso puder ser feito, o software não atende as condições dos termos GPL.

Um *Software Livre* pode ser distribuído com regras restritivas, desde que essas restrições não entrem em conflito com as quatro liberdades. Por exemplo, o ***COPYLEFT*** é uma regra restritiva que garante que essas quatro liberdades sempre existam.

A licença do produto original não pode ser modificada e o usuário deve ter acesso à mesma, na íntegra.

PUC CAMPINAS
Engenharia de Software

General Public License (GPL)

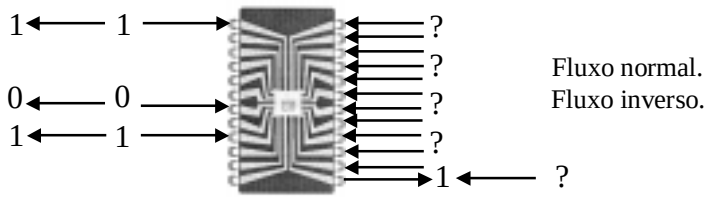
O código fonte deve estar disponível, em conjunto com a versão executável, ou o distribuidor deve informar ao usuário que ele pode adquiri-lo (ou apenas copia-lo sem custo), num período máximo de 3 anos, por um valor não superior ao do meio físico de armazenagem mais custos de reprodução e embalagem.

Não é permitida utilizar-se de partes do código de um software licenciado pela GPL em um software proprietário. Para que isso seja possível, o software proprietário como um todo deverá passar a ser software livre, além de necessitar autorização formal do autor do componente utilizado.

PUC CAMPINAS
Engenharia de Software

2 - Engenharia Reversa em Código Objeto Executável

Quando o produto é de terceiros e não temos acesso ao código fonte, a atividade de Engenharia Reversa fica semelhante àquela executada em hardware (caixa preta): a partir de entradas projetadas obtém-se saídas que serão analisadas, de forma a deduzirmos o processamento de transformação realizado.



PUC CAMPINAS
Engenharia de Software

Reengenharia de Sistemas

Reengenharia de Sistemas:

Compreende uma série de atividades como: inventário das aplicações, reestruturação da documentação, reestruturação de código e dados, engenharia reversa a partir de projeto e do tipo “caixa preta”, com a intenção de criar novas versões dos produtos mais críticos, dotando-as de alta qualidade e melhor manutebilidade.

PUC CAMPINAS
Engenharia de Software

Reengenharia de Sistemas

Reengenharia de Sistemas:

A rigor, a aplicação de conceitos de “Reengenharia”, implica em fazer um novo produto sem copiar características do produto existente (até porque nesse caso teríamos um caso de Engenharia Reversa).

A Reengenharia preconiza estudar o processo e criar um novo produto, com características radicalmente inovadoras (portanto não copiadas do produto existente).

Exercício I

PUC CAMPINAS
Engenharia de Software

Exercício I

1. Quais tipos de manutenção o usuário deve pagar (através de débito ao seu centro de custo ou no contrato de manutenção) e quais tipos o desenvolvedor deve arcar com seu custo?

2. Qual(ais) tipo(s) são normalmente coberto(s) pelos Contratos de Manutenção?

Exercício II

PUC CAMPINAS
Engenharia de Software

Exercício II

1. Usando a tabela a seguir, aplicar o modelo do Pressman para calcular a quantidade de defeitos latentes.

2. O que pode explicar:
   a) O decréscimo do fator de ampliação de erros?
   b) O aumento de erros gerados na fase, em relação às 3 primeiras fases?
   c) O crescimento do percentual de detecção de erros na fase, em relação às 3 primeiras fases?

PUC CAMPINAS
Engenharia de Software

Exercício II

Fases	1	2	3	4	5
Def.Requisitos	0	0	0	20	20
Análise e Projeto	6	10	2,0	30	50
Teste unitário	20	8	1,5	38	60
Teste de integração	0	0	0	0	40
Teste de validação	0	0	0	0	40
Teste de sistema	0	0	0	0	40

PUC CAMPINAS
Engenharia de Software

Exercício II

Legenda da Tabela

1 – Erros passados para a próxima fase, sem ampliação
2 – Erros passados para a próxima fase, com ampliação
3 - Fator de ampliação
4 – Novos erros criados na fase
5 – Percentual de detecção de erros da fase

Observação:
Nas divisões, considere apenas a parte inteira do número resultante.