Engenharia Reversa

UFF – Universidade Federal Fluminense

Graduação em Ciência da Computação

Informática I – 2005/2

Antonio Jorge Sapage da Canhota Junior
Diego Alves de Souza
Diogo dos Santos Moutinho
Felipe Paixão Lohnefink

Definição

A Engenharia reversa é uma atividade que trabalha com um produto existente (um software, uma peça mecânica, uma placa de computador) tentando entender como o mesmo funciona e como é o seu comportamento em diversas circunstancias.

Quando é utilizada?

Fazemos Engenharia Reversa quando temos que trocar um produto com defeito, ou quando queremos conhecer como o mesmo funciona e não temos nenhuma documentação.

Por que a Engenharia Reversa na Informática?

- O sistema foi iniciado há muitos anos (até 20 anos atrás).
- O sistema tem pouca documentação e ela não foi atualizada. O que quer dizer que a documentação descreve um estado anterior do sistema, mas não a configuração atual.
- As pessoas que criaram o sistema deixaram a empresa, ninguém pode explicar muitas decisões que foram tomadas.

Mas o sistema tem que evoluir

- Para ser adaptado a novos computadores (mais barato, mais rápido ou porque ninguém mantém mais os velhos).
- Para ser adaptado a novos softwares (novas bibliotecas, novas linguagem de programação, novas ferramentas).
- Para ser adaptado a novas regras (troca de moeda em todos os países da Europa).

A Engenharia Reversa pode ser de programas como nos exemplos a cima ou de dados. Por exemplo, se queremos construir um editor de texto compatível com o MS-Word, vamos ter que entender a representação que ele usa para ler os documentos MS-Word ou poder salvar documentos nesse formato.

Para software, restrições físicas como as dimensões da tubulação sobre a qual a bomba tem que ser montada, são restrições de interface. O velho programa tinha uma interface específica, e costumava ser chamado de maneira bem definida. O novo programa tem que respeitar a mesma interface.

Exemplos de Engenharia Reversa

- Fora da computação
  - **Tupolev Tu-4** - Em 1945, durante a segunda guerra mundial, três bombardeiros americanos modelo B-29 foram forçados a aterrissar em território russo. Os soviéticos os desmontaram e estudaram. Usaram a engenharia reversa para copiar o bombardeiro nos mínimos detalhes. O resultado foi o bombardeiro Tupolev Tu-4 que voou pela primeira vez em 19 de maio de 1947. A produção em série do bombardeiro começou neste mesmo ano.

Exemplos de Engenharia Reversa

- Na Computaçao
  - **IBM-PC compatível** - A IBM abriu mão da patente de sua plataforma, deixando o caminho livre para qualquer um produzir uma maquina que fosse compatível com o IBM-PC.
  - **Wine** - Programa que funciona como a API do Windows. Permite executar aplicativos desenvolvidos para Windows 3.1X, 9X, NT e 200x no GNU/Linux

Exemplos de Engenharia Reversa

- **Samba** - Software que permite sistemas que não estão rodando o Microsoft Windows a compartilhar arquivos com sistemas que estão. A engenharia reversa foi utilizada para descobrir como o compartilhamento de arquivos do Windows funcionava, para que então computadores que não estivessem com a plataforma Windows pudessem simular este comportamento.

Exemplos de Engenharia Reversa

- **OpenOffice.org** – é um conjunto de aplicativos em OpenSource (código aberto). Está disponível para diferentes plataformas: incluindo Microsoft Windows, Unix, Solaris, Linux e Mac OS X. A Suite é compatível com o Microsoft Office.

Reengenharia

A engenharia reversa consiste em apenas analisar o sistema ou a ferramenta para criar uma representação dela. Já a Reengenharia vai além. Analisa-se o projeto, cria-se uma representação do mesmo e, através dessa representação, monta-se uma nova estrutura que funcione Exatamente como a primeira, mas que não seja meramente uma cópia dela.

Definição de engenharia reversa de software

A engenharia reversa de software consiste em analisar um determinado sistema para criar representações do próprio em um nível mais alto de abstração.

Também pode ser encarada como “Voltar atrás no ciclo de desenvolvimento do software”.

Na prática, existem dois tipos de engenharia reversa de software:
No primeiro caso, o código-fonte já está disponível, mas os aspectos mais globais, talvez documentação escassa ou não válida, tem que ser descobertos.
No segundo caso o código-fonte do software não está disponível, e todos os esforços para descobrir uma possível fonte do código para o software são considerados como engenharia reversa.

OBS: As pessoas que trabalham com engenharia reversa de software estão mais familiarizadas com o segundo caso, chegando até a desconsiderar o primeiro.

Técnicas de engenharia reversa sem o código-fonte

Engenharia reversa de software pode ser efetuada por vários métodos.Três grupos principais da engenharia de software são:

Análise de fluxo de dados: Análise através da observação da troca de informações que envolvem “analisadores de bus” e “pacotes de sniffers” por exemplo, para "ouvir" dentro do bus de um computador ou uma conecção de rede, revelando o tráfico de dados "escondidos". O comportamento dos dados no bus ou na rede podem então ser analisados para produzir uma nova implementação do software que imita o mesmo comportamento. Isto é especialmente utilizado na engenharia reversa de drivers de dispositivos.
Desassemblar: Usando um desassembler, conseguimos obter a linguagem de máquina diretamente do programa. Este código é lido e entendido nos seus próprios termos, apenas com a ajuda de “mneminics” da linguagem de máquina. Isto funciona em qualquer programa de computador, mas pode levar um bom tempo, especialmente para alguém que não esteja acostumado ao código de máquina.
Decompilação: Neste método utiliza-se um decompilador, um programa que tenta recriar o código-fonte em uma linguagem de alto nível, tendo disponível apenas o código de máquina.

Engenharia reversa com código-fonte disponível

Extração das Informações

O primeiro trabalho que se deve fazer é coletar informações sobre o sistema a ser estudado.

Código (análise estática)

A primeira fonte, o código, é a mais usada. Ela permite extrair as informações mais básicas do sistema

- Quais são os componentes básicos do sistema: arquivos, rotinas, tipos, variáveis, classes, etc;

- Relações de definição conectam um componente com seu conteúdo (onde ele se encontra);

- Relações de referência conectam um componente com aqueles que o usam (se uma rotina A chamar uma outra rotina B. A depende de B porque uma modificação na definição de B pode ter conseqüências sobre a execução de A).

Trace de execução (análise dinâmica)

A análise estática pode extrair muitas informações de um programa, mas nem todas. Por exemplo, qual parte de uma instrução IF é realmente usada pode depender dos dados com que o programa foi chamado.

Dados

Os bancos de dados podem ser usados como fonte de informação para ajudar na engenharia reversa de um sistema. Mas a engenharia reversa de dados é também um trabalho específico que pode ser feito independentemente de qualquer sistema que possa manipular esses dados. Por exemplo, poderíamos querer converter um velho banco de dados sobre um “main frame” para um banco de dados relacional e distribuído sobre vários PCs.

Documentação

Chamamos de documentação tudo o que não é usado pelo computador para fazer funcionar o sistema, mas se destina aos engenheiros que usam o código: relatórios, comentários no código, diagramas da análise ou do projeto, etc.

Outras fontes de informação

Finalmente, é possível usar outras fontes de informação. Por exemplo poderíamos procurar quem escreveu cada porção do código. É razoável pensar que se duas partes do código foram desenvolvidas pela mesma pessoa elas tem maior probabilidade de pertencer ao mesmo sub-sistema.

Tratamento das Informações

O objetivo geral dessas atividades é passar as informações do sistema para um nível mais alto de abstração.

Anomalias no código

- Código morto;
- Clones.

Encapsulamento

Em vez de reestruturar um sistema, o encapsulamento propõe esconder o velho código dentro de uma nova camada.
Um bom exemplo de encapsulamento é o programa Wine.

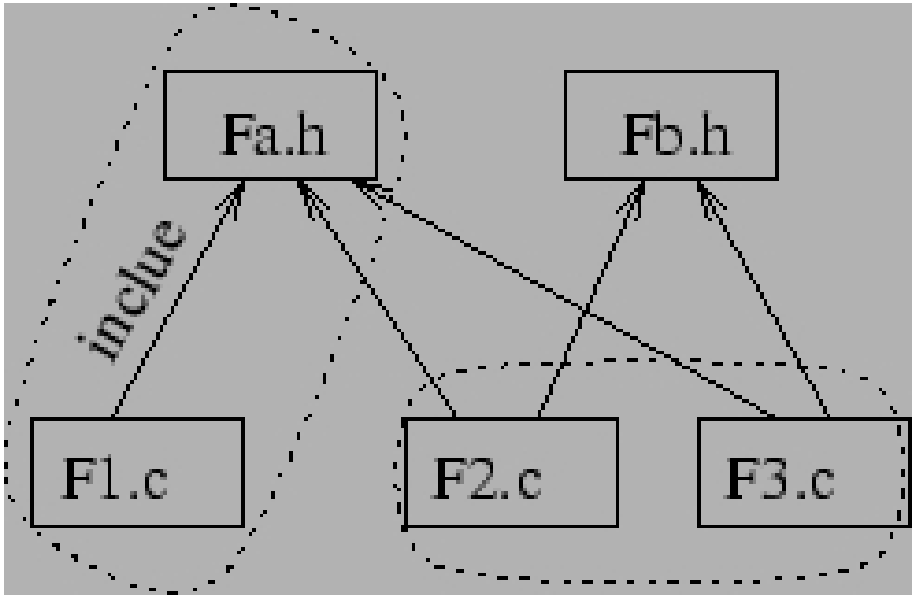
“Slicing” (fatiar)

“Slicing" (fatiar) é uma técnica de decomposição do código de acordo com a utilização das variáveis.

(Re-)Modularização

A (re-)modularização é a decomposição de um conjunto de componentes de software em sub-partes (os módulos).

Para realizar a decomposição são utilizados algoritmos que medem as “distâncias” entre os componentes com base em informações extraídas do sistema.



- O resultado pode ser difícil de entender
- Não há nada mágico neles. Eles agrupam todos os componentes segundo algumas restrições. Se as restrições forem erradas, o resultado também será.
- O resultado desta técnica é globalmente bom.

Reconhecimento de Clichés

- Construção do banco de clichês;
- Dualidade, precisão e cobertura (a descrição do clichê deve ser geral porém corre-se o risco de ocorrerem falso-positivos);
- Tempo de execução;
- Clichês deslocalizados ou interligados (podem existir outras instruções entre as que implementam o clichê ou dois clichês cujas instruções estão misturadas).

Aspectos Legais

- Uma fórmula matemática não pode ser patenteada. Um programa, sendo um algoritmo (portanto, algo matemático) pode ser patenteado?
- Porém, o desenvolvedor de um programa precisa ter seus direitos protegidos.
- São essas duas questões (além de grandes interesses econômicos) que geram grande parte dos conflitos legais na Engenharia Reversa
- A maior parte das leis tentam buscar uma resposta para essas questões
- “Fair Use”
  - Análise de compatibilidade;
  - Correção de erros;
  - Estudos acadêmicos;
  - Evolução tecnológica (com base em outro programa, aprimorá-lo).

Aspectos Legais

- Nos Estados Unidos, “Digital Millenium Copyright Act”
  - Bastante restritiva: somente para fins de analisar compatibilidade
- E na Europa, o “EU Copyright Directive”
  - Inspirado no modelo norte-americano, porém mais flexível: se o objetivo não for lucro ou infração de copyright, é permitido
- Lei Japonesa de Concorrência Desleal de 1993
  - Considera como modelo para futuras leis à respeito de Engenharia Reversa
    - Mesmo que um programa não seja patenteado, a lei o protege
    - Porém ressalva modificações para aperfeiçoamento, padronização, compatibilização
    - O progresso cientifico não é prejudicado, evita-se o monopólio e o direito autoral é protegido

Aspectos Legais

- A lei suíça incentiva, ou melhor, obriga as empresas a praticarem engenharia reversa para diminuir concorrência, mas na prática...
- Na Rússia e Noruega, por exemplo, a Engenharia Reversa é permitida com praticamente nenhuma restrição
- No Brasil, em contrapartida, nao existe nenhuma lei específica sobre o assunto: depende da finalidade

Aspectos Legais

- Dois casos famosos
  - Caso do DVD no Linux
    - Jon Johansen aplica engenharia reversa nas chaves que decodificam DVDs
    - Com isso, ele consegue habilitar o Linux para reproduzir DVDs
    - É acusado pela indústria cinematográfica de “incentivar” a pirataria
  - Borland x Lotus
    - O Lotus 1-2-3 era líder de mercado e a Borland desenvolveu o Quattro Pro com interface idêntica à do rival e, inclusive, com possibilidade de abrir e criar arquivos no formato rival
    - A Lotus processa alegando que houve infração de copyright, usando engenharia reversa
    - A decisão da justiça considerou o processo um “absurdo” e a Borland ganhou