

Noções de Software

Por
José Luís Carneiro



- A rigor, tudo o que pode ser armazenado eletronicamente pode ser chamado de *software*
- Consideraremos aqui, como *software*, apenas os conjuntos de instruções que determinam o comportamento do computador
 - Camada intermediária entre o homem e a máquina, traduzindo os dados para a linguagem de máquina e as informações de volta para a linguagem humana

Pirâmide de *software*



Usuário
final

Software Aplicativo

Linguagens de alto nível

Montadores, interpretadores e compiladores

Software Básico: Sistema operacional + *Drivers*

Máquina convencional – Linguagem de máquina (*Assembly*)

Microprogramação (usando circuitos eletrônicos)

Lógica digital (circuitos eletrônicos)



MÁQUINA

- Programas utilizados pelos usuários
 - Automação de escritório:
 - Editores de texto, planilhas eletrônicas e programas de apresentação
 - Sistemas comerciais e gerenciadores de bancos de dados
 - Sistemas de gestão empresarial:
 - ERP (*Enterprise Resource Planning*)
 - CRM (*Customer Relationship Management*)
 - Projetos (CAD e CAM)
 - Programas gráficos
 - Programas educacionais, utilitários, jogos, etc.

- Programas que fornecem a infra-estrutura para execução dos *softwares* aplicativos
 - BIOS (*Basic Input/Output System*)
Software, gravado numa ROM, que determina como o computador deve se comunicar com os seus diversos periféricos
 - *Drivers*
Pequenos programas que instruem o computador sobre como se comunicar com um determinado periférico. Ampliam as instruções da BIOS e disponibilizam funções mais avançadas
 - Sistemas Operacionais
Conjunto de programas que controla os vários componentes do *hardware*, coordenando as funções básicas do computador, tornando-o operacional. Serve de interface com o usuário. Todo equipamento precisa ter um sistema operacional para funcionar

Comparativo de Sistemas Operacionais

Sistema Operacional	Suporte “Multi-”			Indicado para uso
	Usuários	Tarefas	Processado	
MS-DOS	Não	Não	Não	Doméstico
Windows 3.x	Não	Quase	Não	Doméstico
Windows 9x	Não	Sim	Não	Doméstico
Windows NT	Sim	Sim	Sim	Profissional
Windows 2000	Sim	Sim	Sim	Profissional
Windows XP	Sim	Sim	Sim	Profissional
Unix/Linux	Sim	Sim	Sim	Profissional

- Usadas para escrever os programas que serão executados pelo computador.
 - Baixo nível:
 - Pouco amigáveis
 - Difícil aprendizado
 - Exemplo: *Assembly*
 - Alto nível:
 - Muito amigáveis (próximas à linguagem natural)
 - Fácil aprendizado
 - Exemplo: FORTRAN, COBOL, PASCAL, BASIC, C++, JAVA

- Código-Fonte (ou programa-fonte)
 - O código do programa, na forma em que foi escrito
 - Normalmente um arquivo texto contendo instruções em uma linguagem de programação
 - Precisa ser convertido em binário
- Código-Executável (ou programa-executável)
 - O programa depois de convertido para o formato binário
 - O código fonte é convertido de um arquivo texto compreensível para nós para um arquivo em binário (seqüência de uns e zeros), compreensível para o computador

- Interpretação
 - Linha a linha, o código-fonte é lido, as instruções são convertidas (traduzidas) para binário e executadas pelo computador
 - A cada execução, o processo precisa ser repetido.
- Compilação
 - O código-fonte é lido, todo de uma vez, e convertido (traduzido) para uma seqüência de uns e zeros compreensível para o computador (programa-executável)
 - Uma vez convertido, basta usarmos o programa-executável

Processos de conversão (2)

		Programas	
		Compilados	Interpretados
Vantagens	<ul style="list-style-type: none"> • Não permitem alterações no código-fonte (oferece maior segurança) • Muito mais rápidos que os programas interpretados 	<ul style="list-style-type: none"> • Normalmente permitem alteração no código-fonte (mutabilidade) • Multiplataforma • Tamanho reduzido 	
Desvantagens	<ul style="list-style-type: none"> • Presos à plataforma onde foram compilados 	<ul style="list-style-type: none"> • Mais lentos que os programas-executáveis • Necessitam de um interpretador (<i>runtime</i>) para serem executados 	

- Conjunto de programas aplicativos para realizar tarefas específicas e de uso freqüente.
 - Maneira de reduzir custos e encurtar o prazo de implantação de sistemas aplicativos
 - Desenvolvido de tal forma que possa ser aplicado por um grande número de usuários, ou seja, busca um bom mercado potencial
 - Maior vantagem
 - O pacote será relativamente barato, por dividir seu custo entre vários usuários
 - Maior desvantagem
 - É pouco provável que o pacote atenda a todos os requisitos dos usuários

- Determinam os direitos e deveres para com o *software*
 - *Software* Comercial – Uso condicionado a pagamento prévio (compra)
 - *Demo* (demonstração) – Uso limitado, normalmente para avaliação
 - *Shareware* – Uso liberado para avaliação, o registro é pago
 - *Adware* – Uso condicionado a exibição de propagandas e anúncios
 - *Freeware* – Uso gratuito
 - *Open-source* (código aberto) – Possui direitos autorais, mas o código-fonte é distribuído com o programa
 - *Public domain* (Domínio público) – Sem direitos autorais
 - *Free Software* (*software* livre) – Distribuído sob a licença GPL
“O *software* é livre, o que é diferente de ‘*software* gratuito’”!

- Publicada pela *Free Software Foundation* (FSF)
 - <http://www.fsf.org/home.pt.html>
- O programa pode ser cobrado
 - Conceito distinto do conceito de *software* gratuito
- Código-fonte distribuído com o programa, sem custo adicional
- Melhoras/alterações no código-fonte devem ser repassadas para toda a comunidade
 - http://www.magnux.org/doc/GPL-pt_BR.txt
 - <http://creativecommons.org/licenses/GPL/2.0/>