Capítulo 3
Arquitectura dum Sistema Operativo

SUMÁRIO:

- Componentes dum sistema operativo
- Serviços dum sistema operativo
- Chamadas ao sistema
- Software de sistema
- Estrutura do sistema
- Máquinas virtuais
- Desenho e implementação do sistema
- Criação do sistema

3.1 Componentes dum Sistema Operativo

Enquanto programa, um sistema operativo moderno está estruturado em *subsistemas* ou *módulos* responsáveis por:

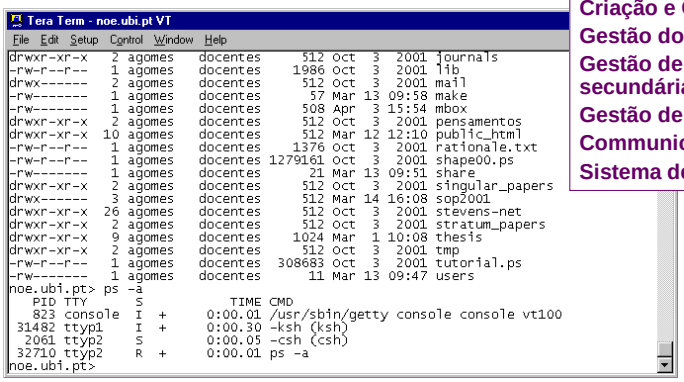
- Gestão de processos
- Gestão de memória principal
- Gestão de ficheiros
- Gestão do sistema de I/O
- Gestão de memória secundária
- Comunicação em Redes
- Sistema de protecção
- Interpretador de comandos

Interpretador de Comandos

- Interpretador de comandos serve de interface entre o SO e o utilizador.
- Alguns SO incluem o interpretador de comandos no próprio núcleo (kernel).
- Outros, tais como o MS-DOS e o Unix, tratam o interpretador de comandos como um programa à parte

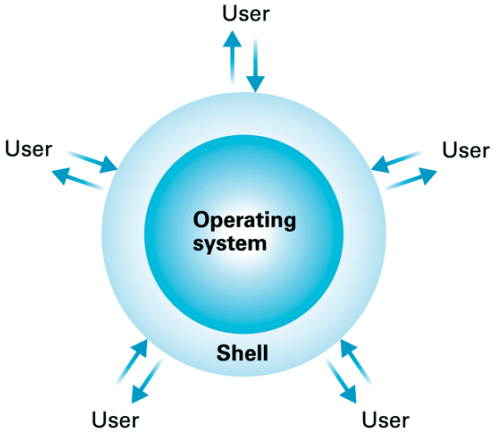
Comandos:

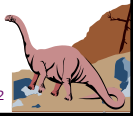
Criação e Gestão de processos
Gestão do sistema de I/O
Gestão de memória principal e secundária
Gestão de ficheiros
Comunicação - Redes
Sistema de protecção



Interpretador de Comandos

Visto com um “shell”





Interpretador de Comandos

- O interpretador de comandos poderá funcionar em modo texto (teclado) ou em modo gráfico (GUI=Graphical User Interface com Rato) ou até em modo reconhecimento de voz
- GUI's – Utilizadores manipulam ***on-screen pictures chamados "icons***" e com "menus" usando alguma tecnica para indicar "selecção" e "escolha" em vez do teclado
- Existem Interpretadores/GUI'S baseados em canetas usados em "hand held computers" (tais como o Newton e Palm) ou até Ecrãs Tácteis (Telemóveise iPad).

Um SO pode ofereçer multiplos interpretador de comandos

Finder: (Primeiro Interpretador Grafico disponivel Comercialmente) Macintosh,

Explorer: Windows 95,98,NT, ..

DOS: - MS-DOS, PC-DOS-2000, Windows 3.1, Windows 95, 98 ..

Bash: (Bourne-again Shell): Linux, BeOS

DCL: OpenVMS

CLI : CISCO IOS

Scripting

Um "shell script" é uma ficheiro de comandos ***interpretados*** em sequencia.
Pode ter associado um linguagem especifico

- command-line shells normalmente oferecem uma infra-estrutura para linguagens de scripting
- É uma maneira mais eficaz de automatizar tarefas de administração
- Permite o rápido desenvolvimento de protótipos de programas
- Os ficheiros de "batch" de windows são um exemplo pobre dum script. Por exemplo através os scripts um servidor Unix pode ser administrado facilmente remotamente (ssh/telnet). Compare com um servidor de Windows NT que não tem esta capacidade nem fornece ferramentas suficientes para isto !
  - Windows powershell veio colmatar esta falha
    - http://en.wikipedia.org/wiki/Windows_PowerShell

Gestão de Processos

- Um *processo* é um programa em execução.
- Um *processo* precisa de certos recursos para completar a sua tarefa, nomeadamente:
  - tempo de CPU
  - memória
  - ficheiros
  - dispositivos de I/O.
- No que respeita a gestão de processos, o sistema operativo é responsável por:
  - Criação, gestão e eliminação de processos.
  - Suspensão (Suspend) e ressunção (Activate) de processos.
  - Provisão de mecanismos para:
    - Sincronização de processos
    - Comunicação entre processos

Gestão de Memória Principal

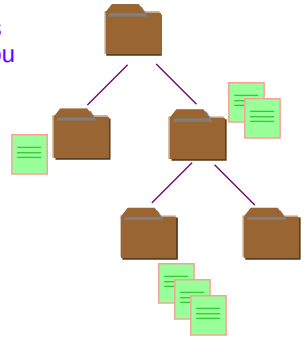
- A execução dum programa requer o seu carregamento prévio em memória. (loader)
- A utilização eficiente do processador e da memória requer o carregamento de vários programas executáveis em memória.(multiprogramação)
- Há vários algoritmos de gestão de memória.
- O sistema operativo é responsável pelas seguintes actividades no que respeita à gestão de memória:
  - Registo actualizado das zonas de memória sob utilização e por quem.
  - Decisão sobre os processos a carregar em memória face ao espaço ainda disponível em mémoria.
  - Reservar e libertar espaço de memória.
  - Defragment

Gestão de Memória Secundária

- Visto que a memória principal (*primary storage*) é volátil e demasiado pequena para armazenar todos os dados e programas duma forma permanente, o computador tem de fornecer memória secundária (*secondary storage*) para salvaguardar (*backup*) a memória principal.
- A maior parte dos computadores modernos usam discos como os principais meios de armazenamento secundário de programas e dados.
- O sistema operativo é responsável pelas seguintes actividades no que respeita à gestão de memória secundária:
  - Gestão de espaço livre
  - Reserva de espaço em disco
  - Escalonamento dos acessos ao disco

Gestão de Ficheiros

- **Ficheiro**: unidade lógica de armazenamento.
- Um ficheiro é uma colecção de informação inter-relacionada e definida por alguém. Normalmente, os ficheiros representam programas (em código fonte ou em código objecto) e dados.
- **Tipos de ficheiros**:
  - ficheiros
  - directorias
- Um dos objectivos dum sistema operativo é proporcionar o acesso uniforme e transparente à informação armazenada.
- O sistema operativo é responsável pelas seguintes actividades no que respeita à gestão de ficheiros:
  - Criação e eliminação de ficheiros.
  - Criação e eliminação de directorias.
  - Manipulação de ficheiros e directorias.
  - Mapeamento ficheiros em memória secundária.
  - Armazenamento de ficheiros em memória secundária.



Sistema de ficheiros

Gestão do Sistema I/O

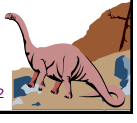
- Um dos objectivos dum sistema operativo é virtualizar o hardware.
- A virtualização do sistema de E/S é feita à custa de device-drivers.

- O sistema I/O é composto pelos seguintes componentes:
  - Um sistema de *buffer* / *caching* / *spooling*
  - ***s**imultaneous **p**eripheral **o**peration **o**n-**l**ine – Operação de Disco*
  - Uma interface geral para device-drivers
  - Device-drivers para dispositivos específicos de hardware

- A virtualização do sistema em UNIX – cada dispositivo de I/O é um ficheiro !

Sistema de Protecção

- *Protecção* é um mecanismo para controlar o acesso ao sistema e aos seus recursos face a processos e utilizadores.
- O sistema deve ter mecanismos para definir / alterar / gerir o controlo de acesso a cada recurso do sistema.
- O mecanismo de protecção tem de distinguir entre utilização autorizada e não-autorizada.
- O sistema operativo deve garantir uma utilização autorizada de ficheiros, memória, CPU, entre outros recursos do sistema, nomeadamente:
  - acesso limitado ao espaço de endereçamento dum processo
  - autorização para envio de sinais a processos
  - operações de E/S executadas exclusivamente pelo sistema



Redes (Sistemas Distribuídos)

- Um *sistema distribuído* é uma colecção de processadores que não partilham memória nem relógio. Cada processador tem a sua própria memória local.
- Os processadores no sistema estão ligados através duma rede de comunicações.
- A comunicação tem lugar através da utilização dum *protocolo.*
- O sistema operativo moderno disponibiliza normalmente serviços / primitivas para comunicação com outros computadores.
- Nalguns sistemas operativos, o acesso à rede de dados é vista como uma generalização do acesso a ficheiros.
- Portanto, existe também um device-driver de rede que oculta os detalhes de funcionamento da rede.

3.2 Serviços do Sistema Operativo

- Um SO proporciona um ambiente próprio à execução de programas**.**
- Para isso, o SO. oferece ao programador um conjunto de **serviços** que visam facilitar a programação, nomeadamente:

chamadas de API (Application Programming Interface)

- **Execução de programas** – capacidade do sistema para carregar um programa em memória e executá-lo.
- **Operações I/O**– dado que os programas do utilizador não podem executar operações I/O directamente, o sistema operativo tem de fornecer meios por forma a fazê-lo.
- **Manipulação dos sistema de ficheiros**– capacidade dum programa para ler, escrever, criar e eliminar ficheiros.
- **Comunicações** – troca de informação entre processos em execução no mesmo computador ou em computadores diferentes ligados em rede. Comunicação implementada via *shared memory* ou *message passing*.
- **Detecção de erros** – garantia da computação correcta através da detecção de erros na CPU e hardware de memória, em dispositivos I/O, ou em programas do utilizador.

Serviços de Garantia de Eficiência do Sistema Operativo

- Há funções adicionais do sistema operativo não para ajudar o utilizador, mas para assegurar a eficiência das operações do sistema:
  - **Reserva de recursos** – reserva de recursos para vários utilizadores ou processos em execução simultânea.
  - **Inventário** – regista a quantidade e os tipos de recursos usados pelos utilizadores. O inventário é importante para fins estatísticos, segurança, despesas de utilização de recursos, etc.
  - **Protecção** – assegurar que o acesso a todos os recursos do sistema é controlado.

Chamadas ao Sistema

- Chamadas ao sistema fornecem uma **interface** entre um processo e o sistema operativo.
  - Geralmente disponíveis como instruções em linguagem assembly.
  - Linguagens (e.g., C, C++) podem substituir a linguagem assembly para efectuar chamadas directas ao sistema
- Normalmente as chamadas ao sistema são em tudo idênticas a vulgares chamadas a funções.

All Unix variants provide a well defined limited number of **entry points i**nto the kernel called system calls. Unix version 7 about 50, Unix 4.3+BSD about 110 and Unix SVR4 about 120

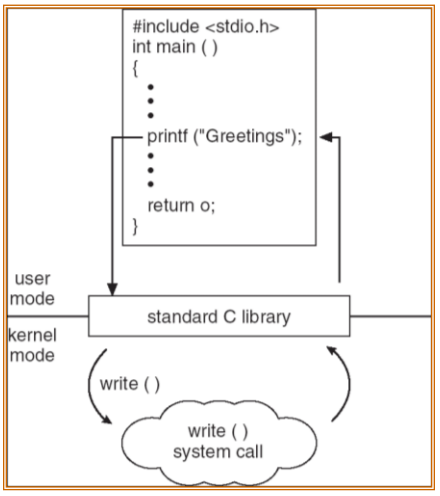
For example printf might invoke the write system call to perform output but the function strcpy and atoi dont involve the OS at all

Page 20 W.R.Stevens “Advanced Prog. In the Unix Env.

Execute process explorer

Standard C Library Example

- Um programa em C que invoke a função printf() da biblioteca padrão <stdio.h>, que faz uma chamada à função de baixo nível de sistema write()



Abstracção do Disco

Programador de SO

```
load(…);
seek(…);
out(…);
```

(a) Controlo Directo

Programador de Aplicação

```
void write() {
  load(…);
  seek(…)
  out(…)
}
```

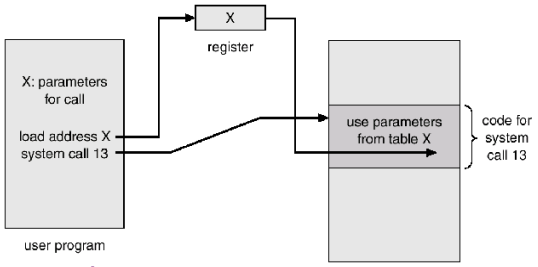
(b) `write()` abstraction

```
int fprintf(…) {
  ...
  write(…)
  …
}
```

(c) `fprintf()` abstraction

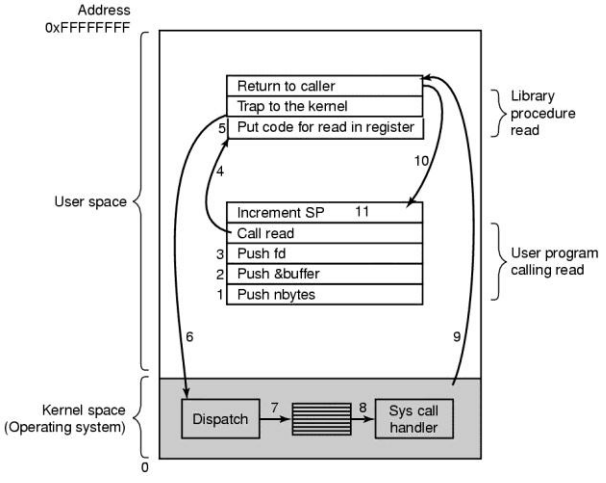
Chamadas ao Sistema

- Há três métodos gerais de passagem de parâmetros entre um processo e o sistema operativo:
  - Passagem de parâmetros através de registos (*registers*).
  - Armazenamento de parâmetros numa tabela em memória, passando o endereço da tabela como parâmetro num registo.
  - O processo armazena os parâmetros na *stack*, e o sistema operativo retira-os da *stack*.



Passagem de parâmetros via tabela

System Calls



The 11 steps in making the system call read(fd, buffer, nbytes). (tannenbaum)

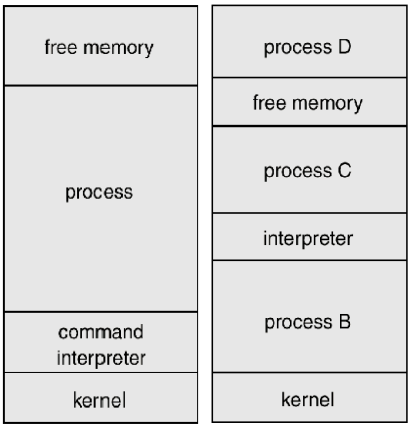
Tipos de Chamadas ao Sistema

Há 5 classes de chamadas ao sistema:

- Controlo de processos
- Manipulação de ficheiros
- Manipulação de dispositivos
- Manutenção de informação
- Comunicação

Controlo de processos

As principais chamadas ao sistema relacionadas com a gestão de processos são:



- **end, abort**
- **load, execute**
- **create, terminate**
- **get/set process attributes**
- **wait for time / event**
- **wait for event**
- **signal event**
- **allocate/free memory**

Malloc-See Stevens Page 21

Manipulação de ficheiros

As principais chamadas ao sistema relacionadas com a manipulação de ficheiros são:

- **create/delete file**
- **open, close**
- **read, write, reposition**
- **get/set file attributes**

Fazer Programa cat

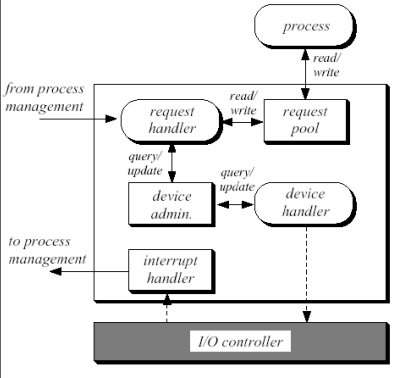
putchar/getchar

Versus

Lowlevel read/write

Manipulação de dispositivos

As principais chamadas ao sistema relacionadas com a manipulação de dispositivos são:



- **request/release device**
- **read, write, reposition**
- **get/set device attributes**
- **attach/detach device**

Manutenção de informação

As principais chamadas ao sistema relacionadas com a manutenção de dispositivos são:

- **get/set time or date**
- **get/set system data**
- **get/set process, file or device attributes**

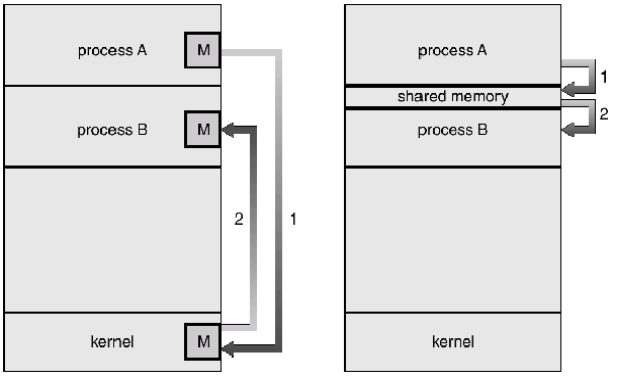
Comunicação

As principais chamadas ao sistema relacionadas com a comunicação são:

- **create, delete communication connection**
- **send, receive messages**
- **transfer status information**
- **attach, detach remote devices**

Modelos de Comunicação

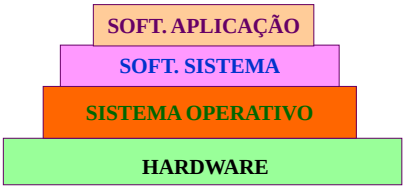
- Há dois modelos de comunicação entre processos:
  - passagem de mensagens
  - memória partilhada.



Message Passing **Shared Memory**

Software de Sistema

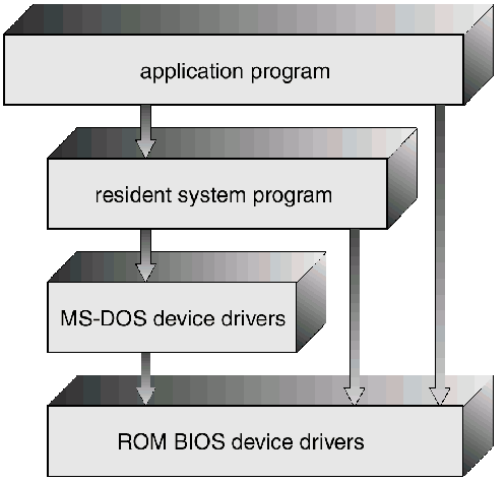
Há uma hierarquia lógica num computador:



- Objectivos de **software de sistema**:
  - ambiente para o desenvolvimento e execução de programas
  - interface para as chamadas ao sistema
- Há 6 categorias de software de sistema:
  - Manipulação de ficheiros (**ex.:** `cp, mkdir`)
  - Informação de estado (**ex.:** `ps, df`)
  - Edição de ficheiros (**ex.:** `pico, vi, joe`)
  - Suporte à programação (**ex.:** `cc, pc`)
  - Carregamento e execução de programas (**ex.:** `ld, csh`)
  - Comunicações (**ex.:** `telnet, mail`)

Estrutura dum Sistema Operativo (MS-DOS)

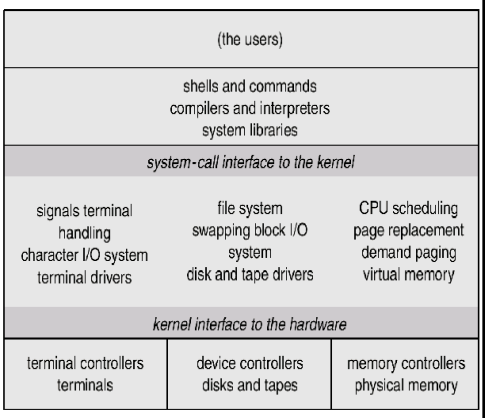
- Um sistema operativo é normalmente desenhado e construído por módulos. Não é pois um sistema monolítico. Portanto, temos de definir para cada módulo, a sua função, as suas entradas e as suas saídas.
- MS-DOS – escrito para fornecer a máxima funcionalidade no mínimo espaço.
  - Não dividido em módulos
  - Embora o MS-DOS tenha alguma estrutura, as suas interfaces e níveis de funcionalidade não estão bem separadas.



Estrutura em camadas do MS-DOS

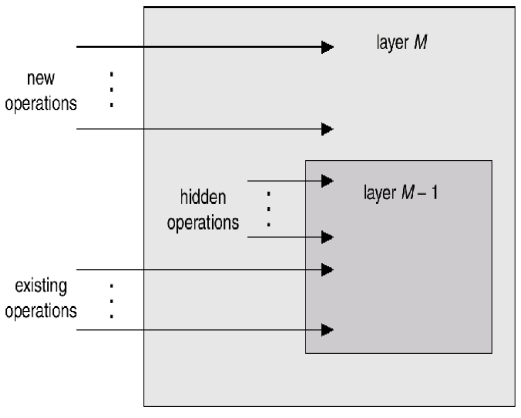
Estrutura dum Sistema Operativo (UNIX)

- UNIX – limitado pela funcionalidade do hardware; o UNIX original tinha uma estruturação limitada. O sistema UNIX consiste em duas partes separadas:
  - Interface de chamadas ao sistema
  - Núcleo (*kernel*)
    - Consiste em tudo o que fica abaixo da interface de chamadas ao sistema e acima do hardware.
    - Fornece o sistema de ficheiros, escalonamento da CPU, gestão de memória, e outras funções do sistema operativo.



Abordagem por Camadas

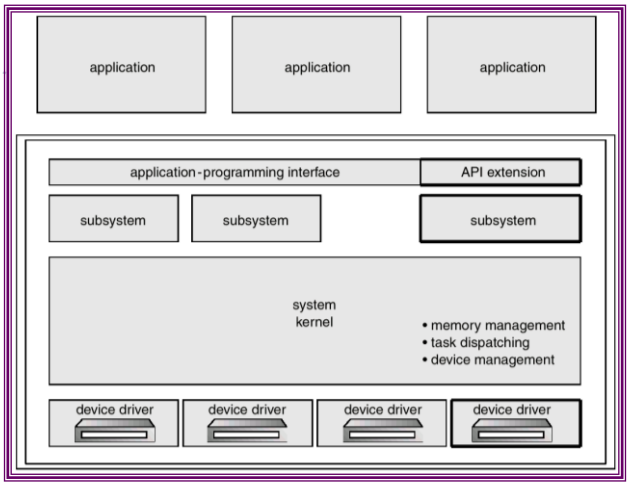
- O sistema operativo está dividido em camadas (*layers*). A camada inferior (*layer* 0) é o hardware; a camada de topo (*layer* N) é a interface com o utilizador.
- Com modularidade, as camadas são seleccionadas tal que cada uma usa funções (operações) e serviços de camadas estritamente abaixo.



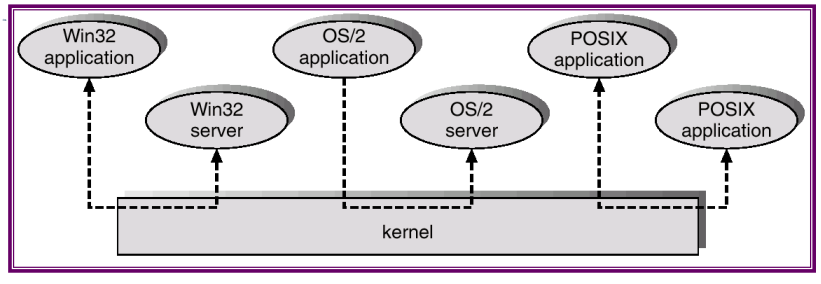
Um sistema operativo por camadas

Fim de Capítulo

OS/2 Layer Structure



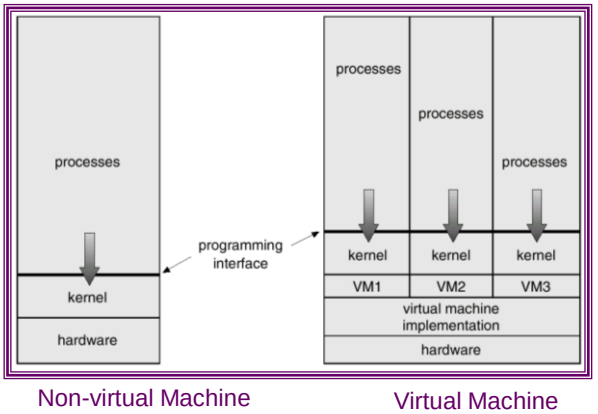
Windows NT Client-Server Structure



Virtual Machines

- A *virtual machine* takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware.
- A virtual machine provides an interface *identical* to the underlying bare hardware.
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.

System Models



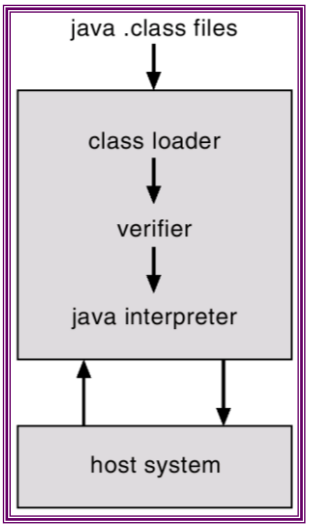
Non-virtual Machine Virtual Machine

Advantages/Disadvantages of Virtual Machines

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating-systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an *exact* duplicate to the underlying machine.

Java Virtual Machine

- Compiled Java programs are platform-neutral bytecodes executed by a Java Virtual Machine (JVM).
- JVM consists of
  - class loader
  - class verifier
  - runtime interpreter
- Just-In-Time (JIT) compilers increase performance



System Design and Implementation

- Goals
  - User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast.
  - System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient.
- Implementation
  - Traditionally written in assembly language, operating systems can now be written in higher-level languages.
  - Code written in a high-level language:
    - can be written faster.
    - is more compact.
    - is easier to understand and debug.
    - An operating system is far easier to port (move to some other hardware) if it is written in a high-level language.
  - Operating systems are designed to run on any of a class of machines; the system must be configured for each specific computer site. SYSGEN program obtains information concerning the specific configuration of the hardware system.

Others

- http://www.slideshare.net/wx672/os-5463020