UFFS
Ciência da Computação

Sistemas Operacionais

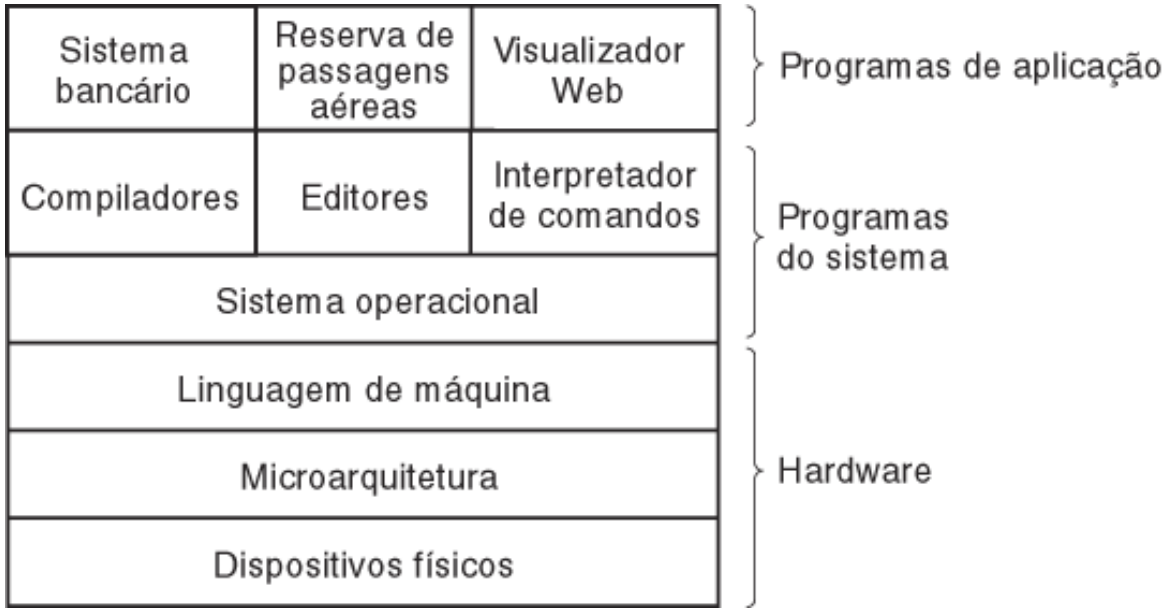
Prof. Marco Aurélio Spohn

Capítulo 1

Introdução

1.1 O que é um sistema operacional
1.2 História dos sistemas operacionais
1.3 O “zoológico” de sistemas operacionais
1.4 Revisão sobre *hardware* de computadores
1.5 Conceitos sobre sistemas operacionais
1.6 Chamadas de sistema
1.7 Estrutura de sistemas operacionais

Introdução

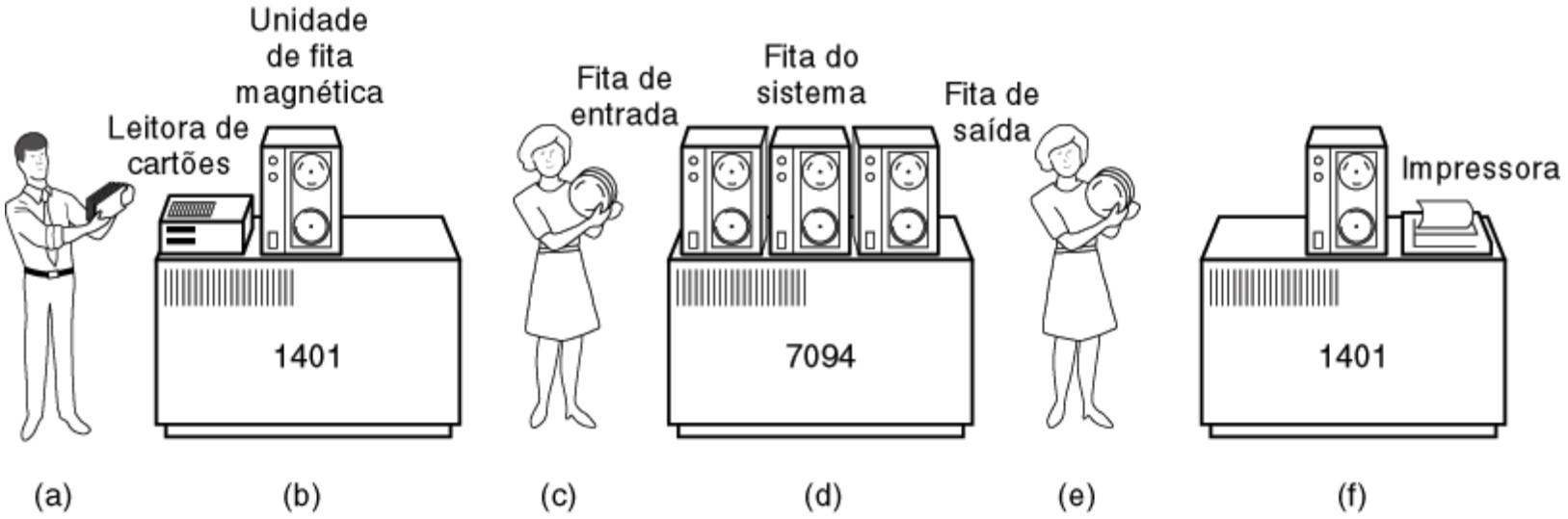


- Um sistema computacional consiste em
  - *hardware*
  - programas do sistema
  - programas de aplicação

O que é um Sistema Operacional

- É uma máquina estendida
  - Oculta os detalhes complicados que têm quer ser executados
  - Apresenta ao usuário uma máquina virtual, mais fácil de usar
- É um gerenciador de recurso
  - Cada programa tem um tempo com o recurso
  - Cada programa tem um espaço no recurso

História dos Sistemas Operacionais (1)



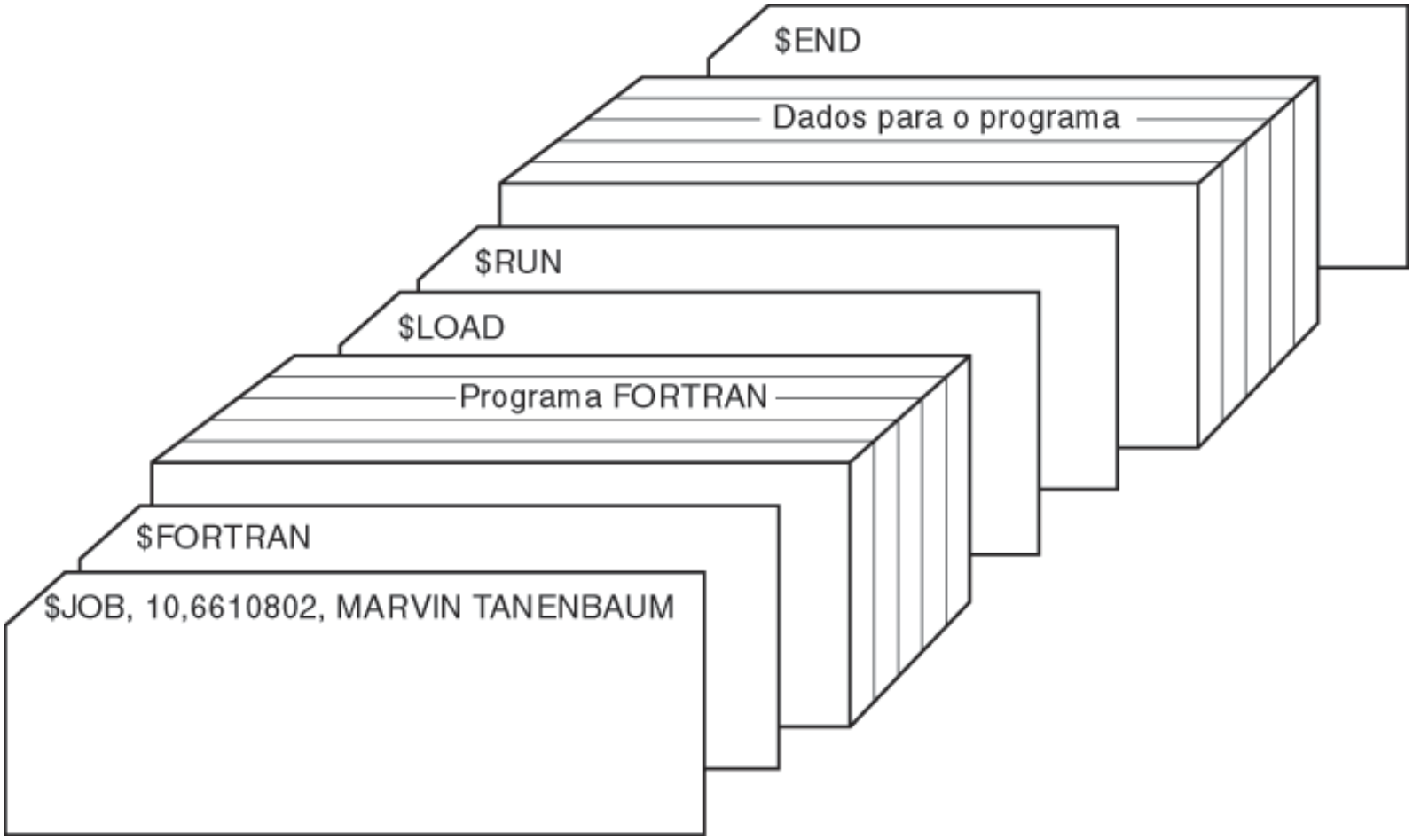
Antigo sistema em lote (*batch*)

- traz os cartões para o 1401
- lê os cartões para a fita
- coloca a fita no 7094 que executa o processamento
- coloca a fita no 1401 que imprime a saída

História dos Sistemas Operacionais (2)

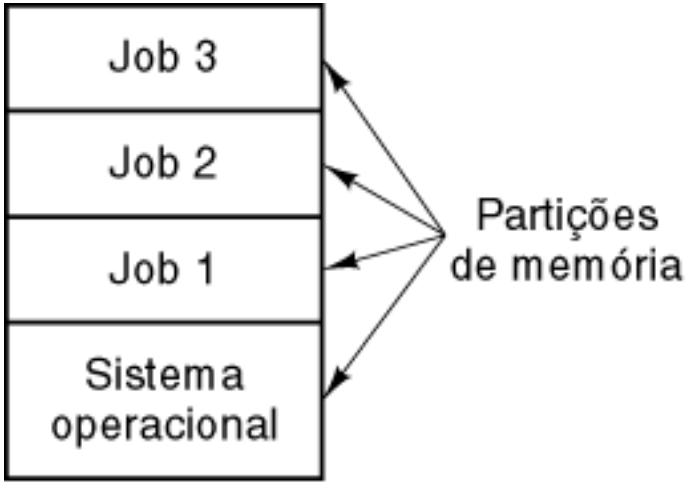
- Primeira geração 1945 - 1955
  - Válvulas, painéis de programação
- Segunda geração 1955 - 1965
  - transistores, sistemas em lote
- Terceira geração 1965 – 1980
  - CIs e multiprogramação
- Quarta geração 1980 – “presente”
  - Computadores pessoais
- Quinta geração (presente):
  - Computação em nuvem (*cloud computing*)

História dos Sistemas Operacionais (3)



Estrutura de um job FMS (*Fortran Monitor System*) típico – 2a. geração

História dos Sistemas Operacionais (4)

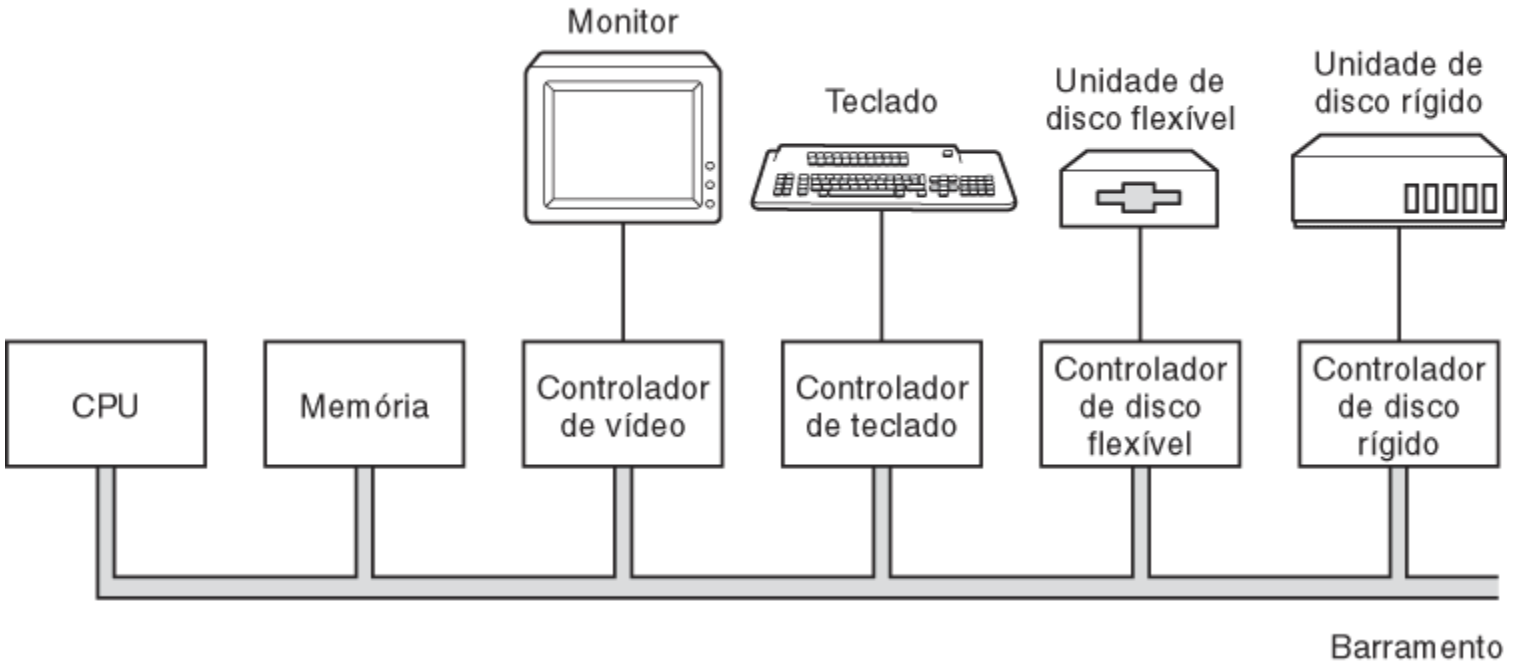


- Sistema de multiprogramação
  - Três *jobs* na memória – 3a. geração

O Zoológico de Sistemas Operacionais

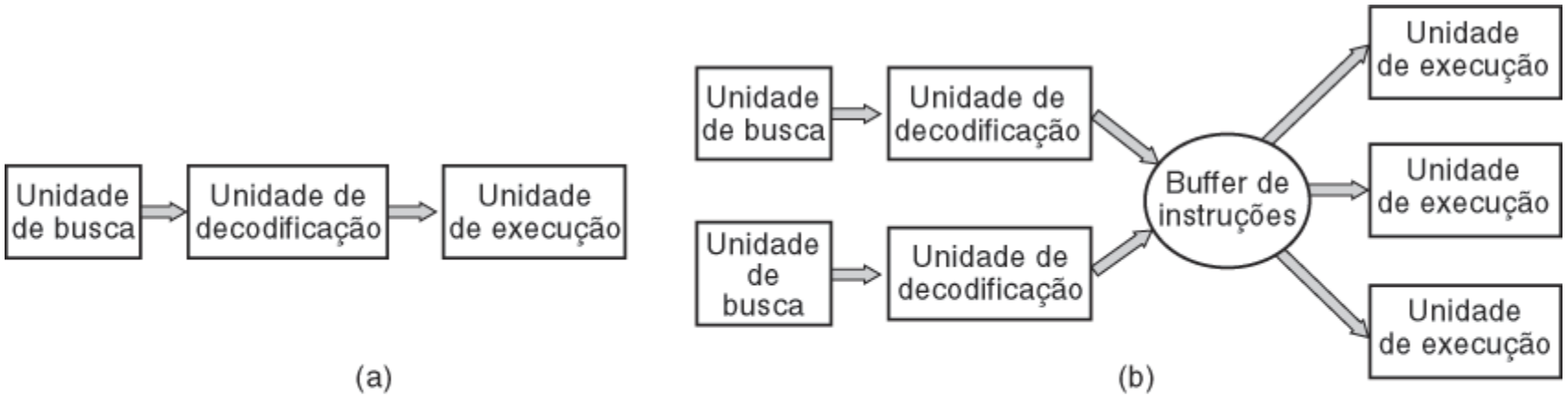
- Sistemas operacionais de computadores de grande porte
- Sistemas operacionais de servidores
- Sistemas operacionais de multiprocessadores
- Sistemas operacionais de computadores pessoais (já faz um certo tempo que também são *multi cores*)
- Sistemas operacionais de tempo real
- Sistemas operacionais embarcados

Revisão sobre *hardware* de computadores (1)



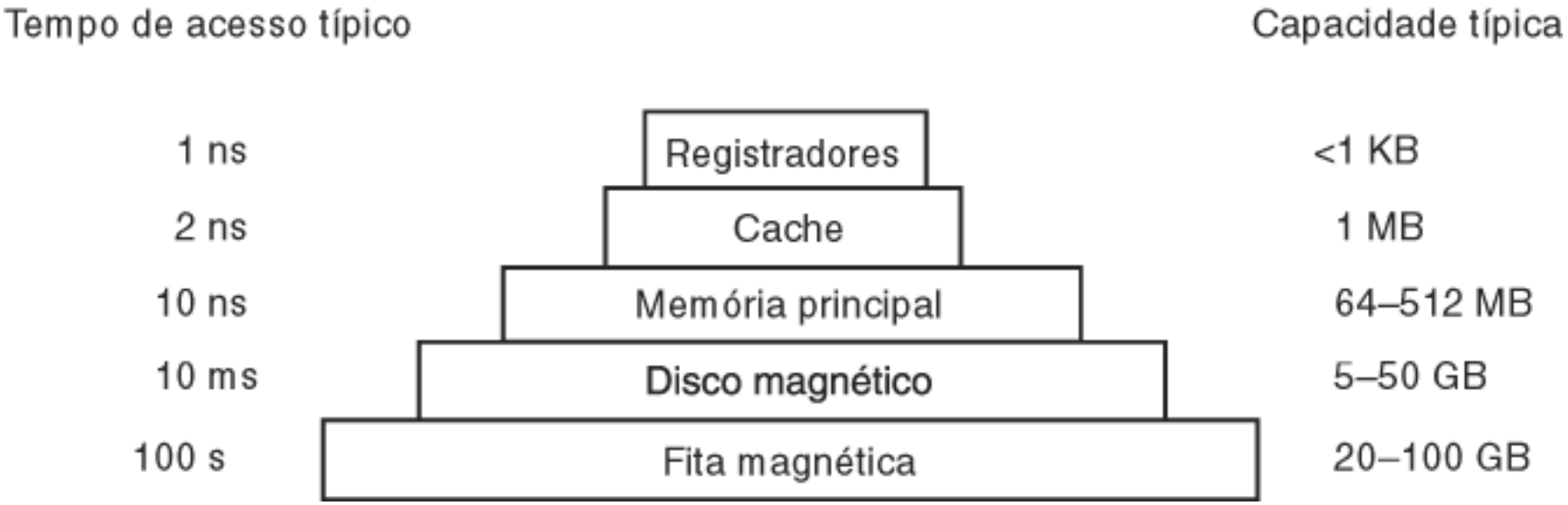
Componentes básicos de um computador pessoal

Revisão sobre *hardware* de computadores (2)



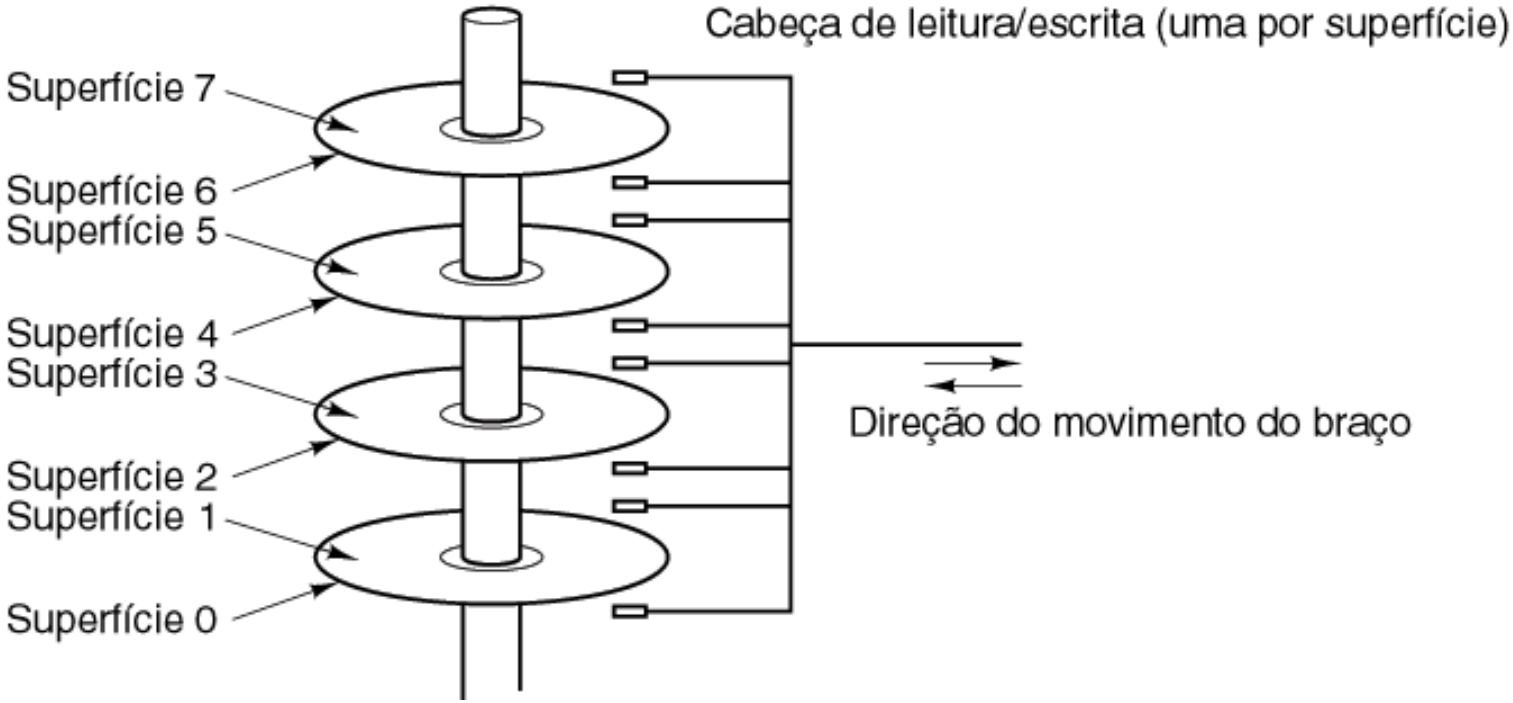
(a) Um *pipeline* de três estágios
(b) Uma CPU superescalar

Revisão sobre *hardware* de computadores (3)



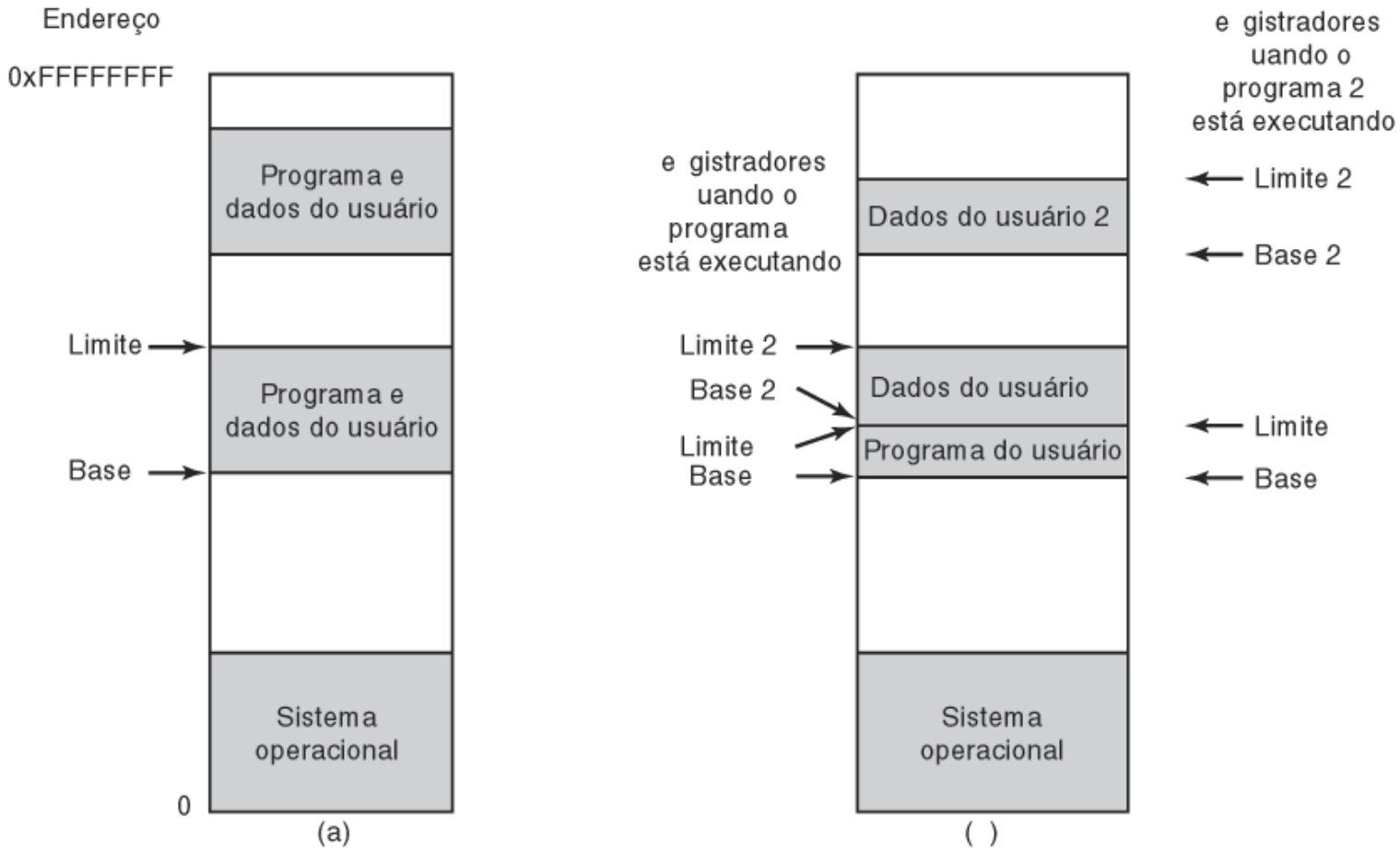
- Típica hierarquia de memória
  - números mostrados são apenas aproximações

Revisão sobre *hardware* de computadores(4)



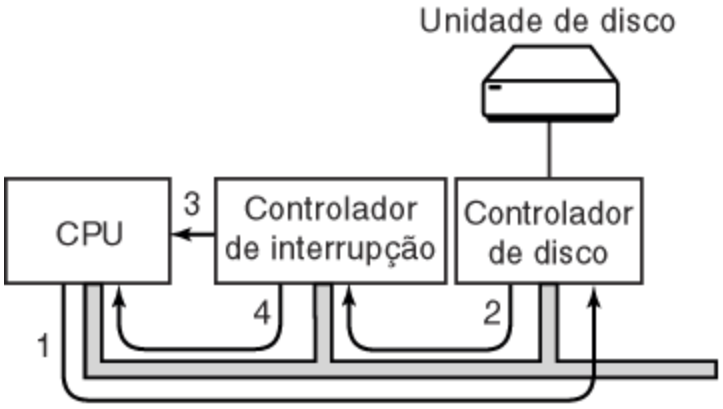
Estrutura de uma unidade de disco

Revisão sobre *hardware* de computadores (5): proteção de memória

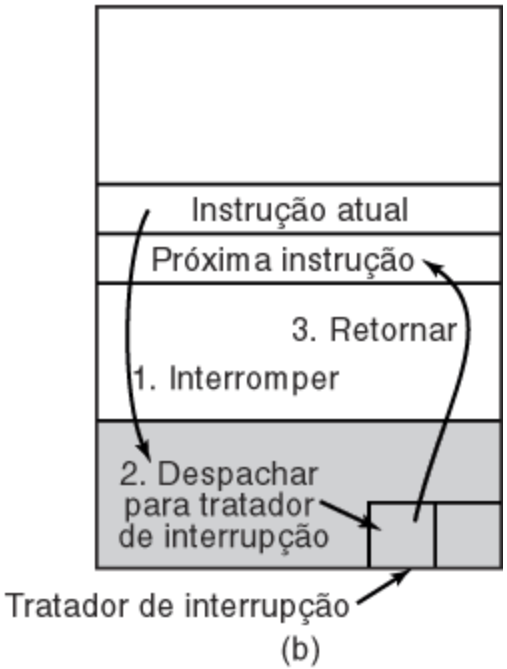


Um par (base, limite) e dois pares (base, limite)

Revisão sobre *hardware* de computadores (6)



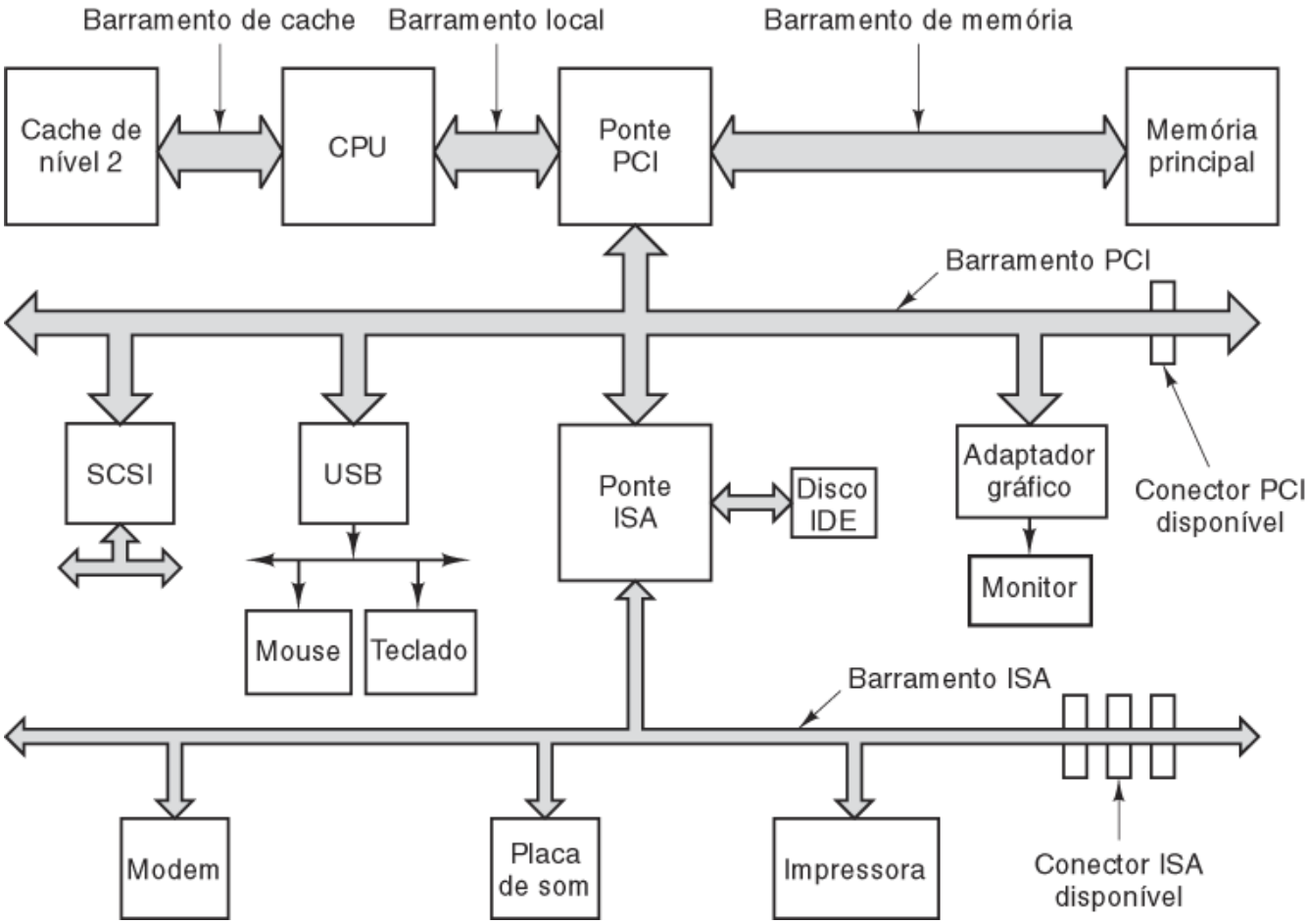
(a)



(b)

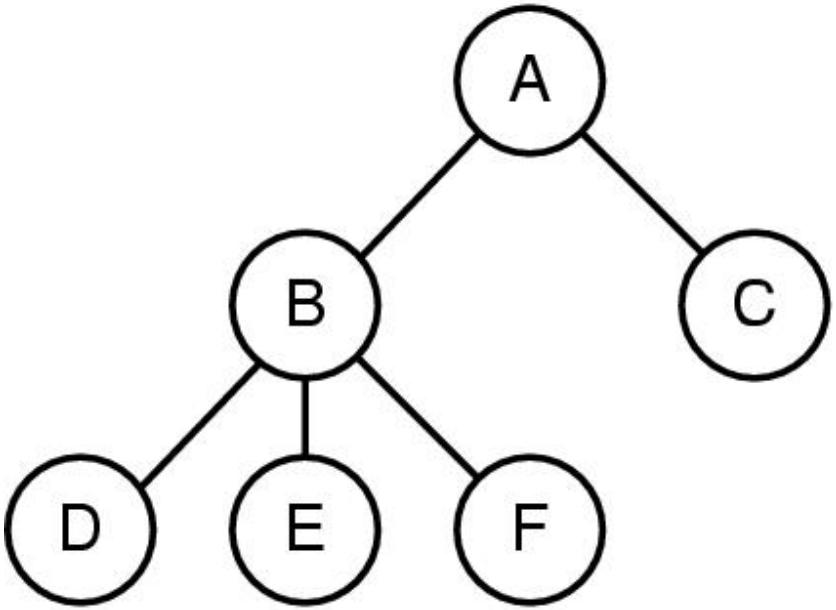
a) Passos para iniciar um dispositivo de E/S e obter uma interrupção
b) Como a CPU é interrompida

Revisão sobre *hardware* de computadores(7)



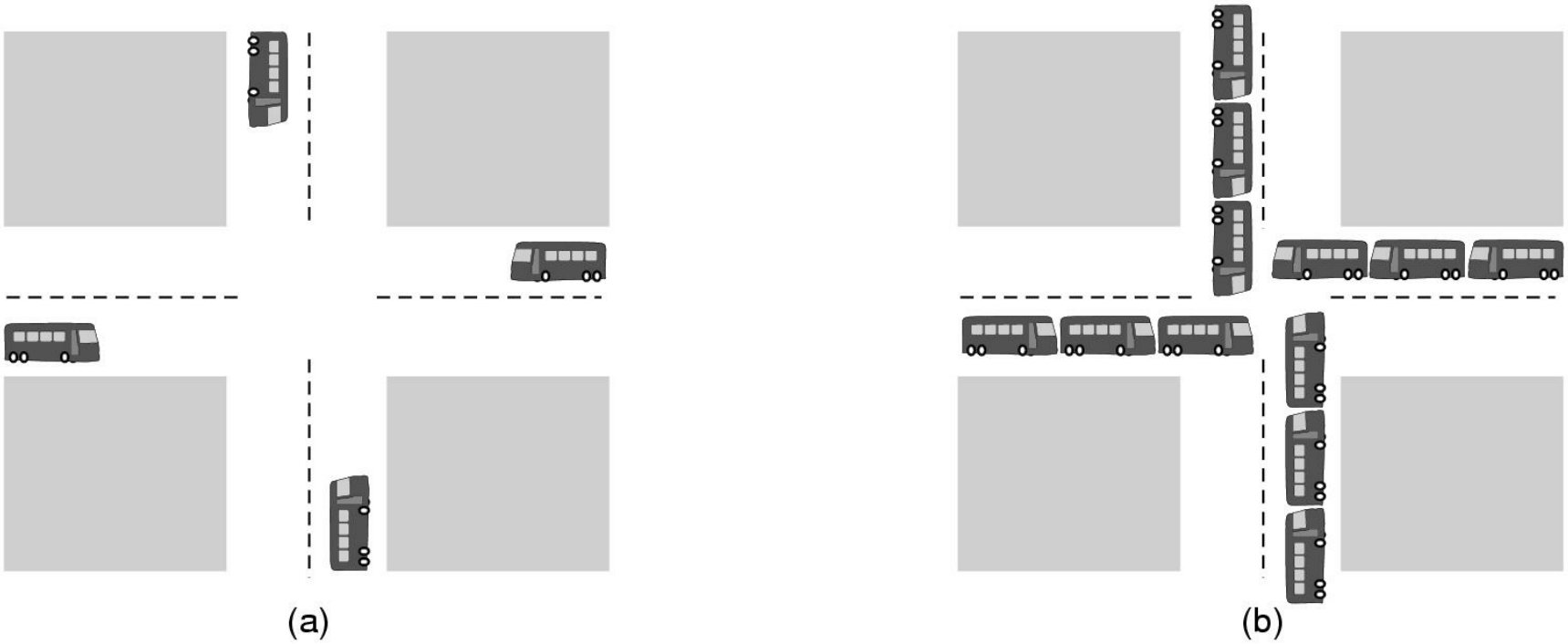
Estrutura de um sistema *Intel Pentium*

Conceitos sobre Sistemas Operacionais (1)



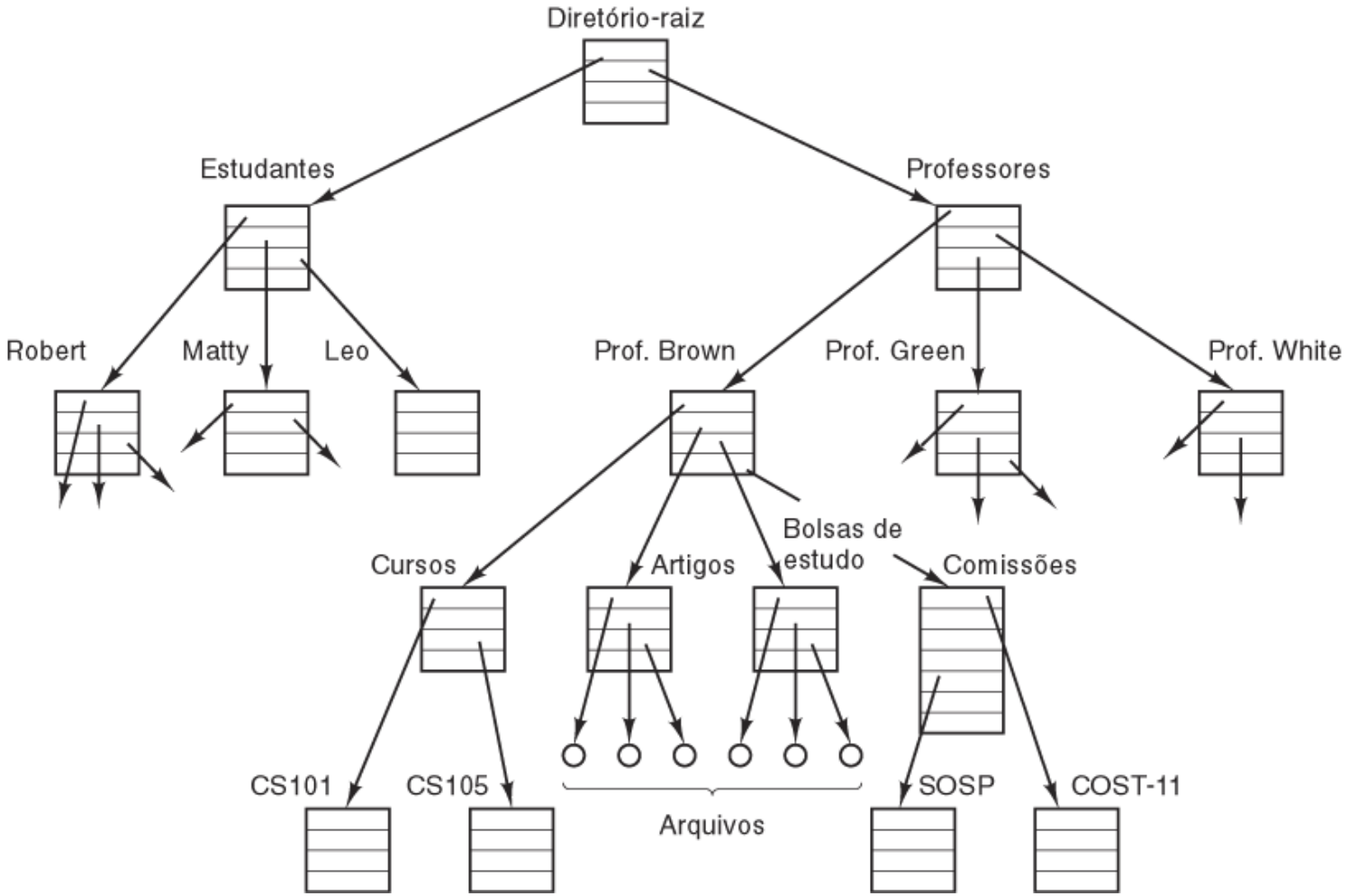
- Uma árvore de processos
  - A criou dois processos filhos: B e C
  - B criou três processos filhos: D, E, e F

Conceitos sobre Sistemas Operacionais (2): impasses



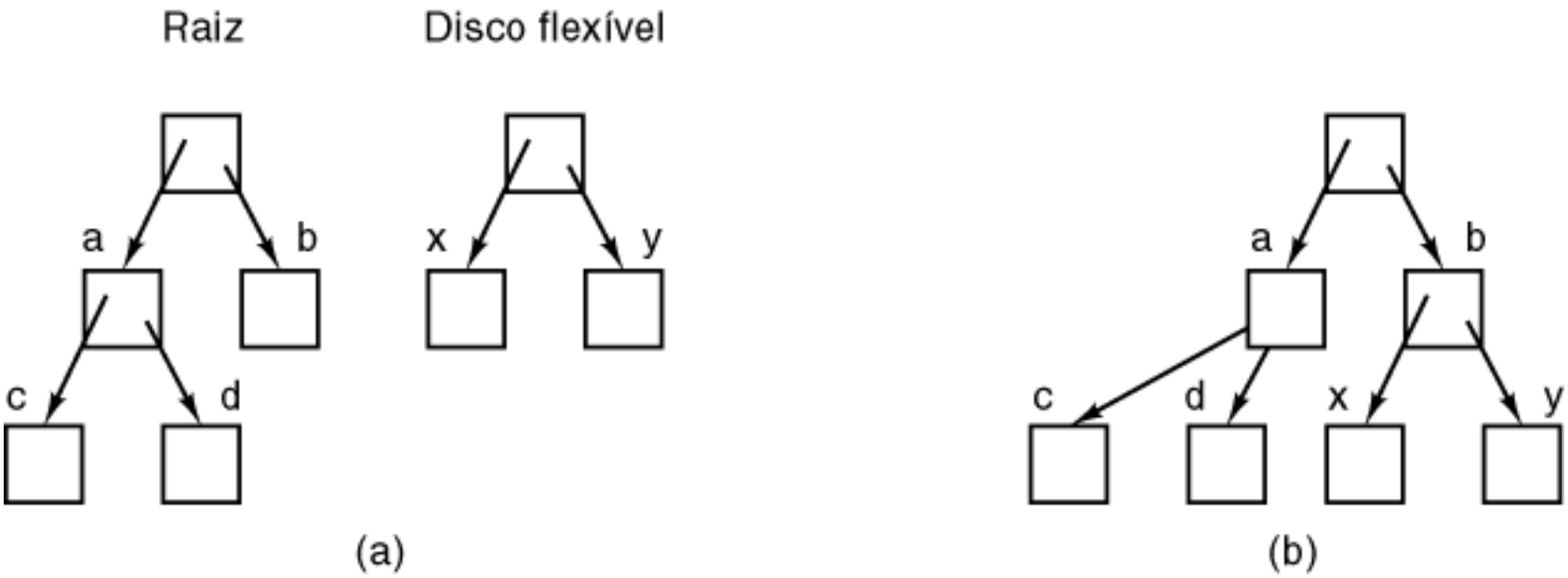
(a) Um *deadlock* potencial. (b) um *deadlock* real.

Conceitos sobre Sistemas Operacionais (3)



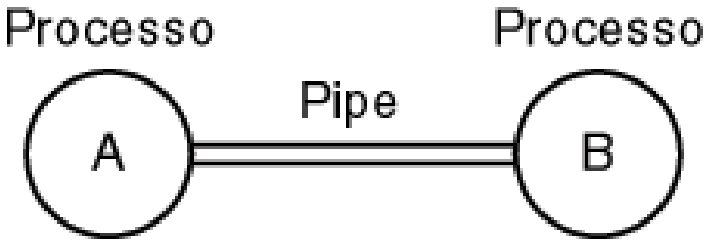
Sistema de arquivos de um departamento universitário

Conceitos sobre Sistemas Operacionais (4)



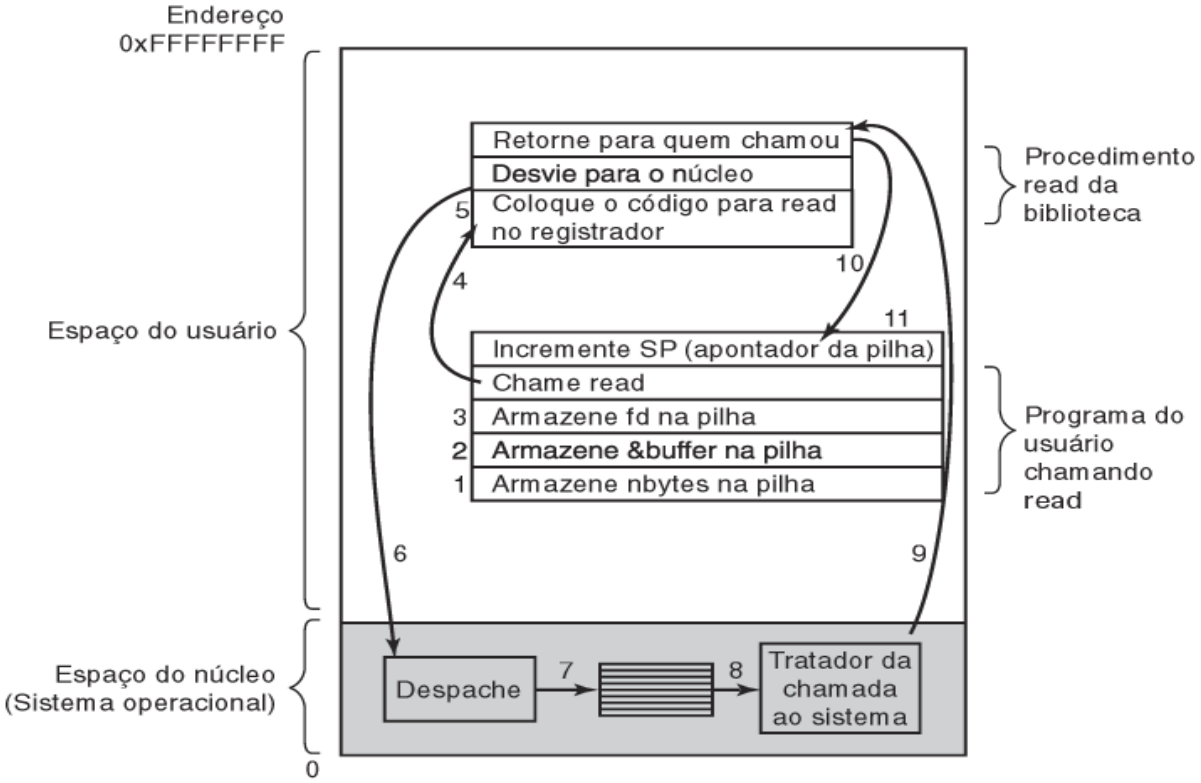
- Antes da “montagem”,
  - os arquivos do disco flexível são inacessíveis
- Depois da “montagem” do disco flexível em b,
  - os arquivos do disco fazem parte da hierarquia de arquivos

Conceitos sobre Sistemas Operacionais (5): comunicação entre processos



Dois processos conectados por um *pipe*

Os Passos de uma Chamada ao Sistema



Passos para fazer uma chamada ao sistema *read (fd, buffer, nbytes)*
O código da *system call* é colocado em um registrador esperado pelo S.O. (passo 5). Para desviar ao núcleo (passo 6), modo privilegiado, faz-se uso de interrupções de *software* (*traps*), tratadas da mesma forma que interrupções de *hardware*. Nesse ponto, desvia-se a execução para o *kernel* de fato. Após a execução da chamada, retorna-se o controle ao processo (caso esteja bloqueado, o mesmo é desbloqueado).

Algumas Chamadas ao Sistema para Gerenciamento de Processos

Gerenciamento de processos

Chamada	Descrição
pid = fork( )	Crie um processo filho idêntico ao processo pai
pid = waitpid(pid, &statloc, options)	Aguarde um processo filho terminar
s = execve(name, argv, environp)	Substitua o espaço de endereçamento do processo
exit(status)	Termine a execução do processo e retorne o estado

Algumas Chamadas ao Sistema para Gerenciamento de Arquivos

Gerenciamento de arquivos

Chamada	Descrição
fd = open(file, how, ...)	Abra um arquivo para leitura, escrita ou ambas
s = close(fd)	Feche um arquivo aberto
n = read(fd, buffer, nbytes)	Leia dados de um arquivo para um buffer
n = write(fd, buffer, nbytes)	Escreva dados de um buffer para um arquivo
position = lseek(fd, offset, whence)	Mova o ponteiro de posição do arquivo
s = stat(name, &buf)	Obtenha a informação de estado do arquivo

Algumas Chamadas ao Sistema para Gerenciamento de Diretório

Gerenciamento do sistema de diretório e arquivo

Chamada	Descrição
s = mkdir(name, mode)	Crie um novo diretório
s = rmdir(name)	Remova um diretório vazio
s = link(name1, name2)	Crie uma nova entrada, name2, apontando para name1
s = unlink(name)	Remova uma entrada de diretório
s = mount(special, name, flag)	Monte um sistema de arquivo
s = umount(special)	Desmonte um sistema de arquivo

Algumas Chamadas ao Sistema para Tarefas Diversas

Diversas

Chamada	Descrição
s = chdir(dirname)	Altere o diretório de trabalho
s = chmod(name, mode)	Altere os bits de proteção do arquivo
s = kill(pid, signal)	Envie um sinal a um processo
seconds = time(&seconds)	Obtenha o tempo decorrido desde 1º de janeiro de 1970

Chamadas ao Sistema (1)

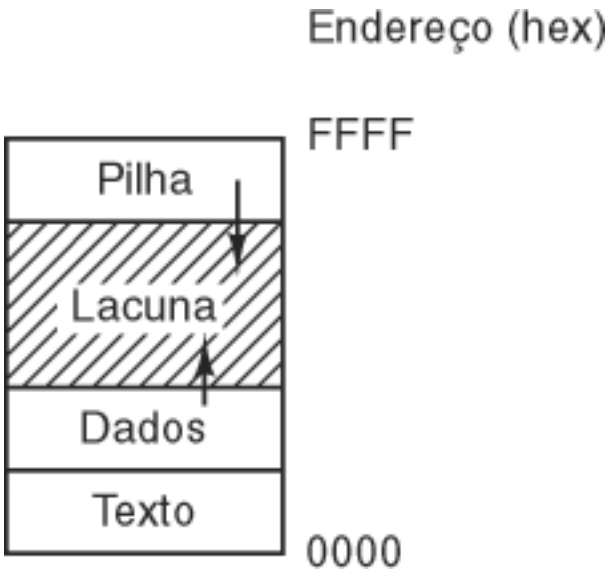
- O interior de um *shell*:

```
#define TRUE 1

while (TRUE) {                                          /* repita para sempre */
        type_prompt( );                                 /* mostra prompt na tela */
        read_command(command, parameters);              /* lê entrada do terminal */

        if (fork( ) !=0) {                              /* cria processo filho */
             /* Parent code. */
             waitpid(–1, *status, 0);                   /*aguarda o processo filho acabar */
        } else {
             /* Child code. */
             execve(command, parameters, 0);            /*executa o comando */
        }
}
```

Chamadas ao Sistema (2)



Os processos têm, no mínimo, três segmentos: texto (instruções), dados e pilha

Em outros sistemas, incluindo o *xv6*, há também um quarto segmento: o *Heap* (espaço para alocação dinâmica de memória)

Chamadas ao Sistema (3): ligação (*link*) ou atalho

a) Dois diretórios antes da ligação de */usr/jim/memo* ao diretório *ast*

b) Os mesmos diretórios depois dessa ligação

Chamadas ao Sistema (4)

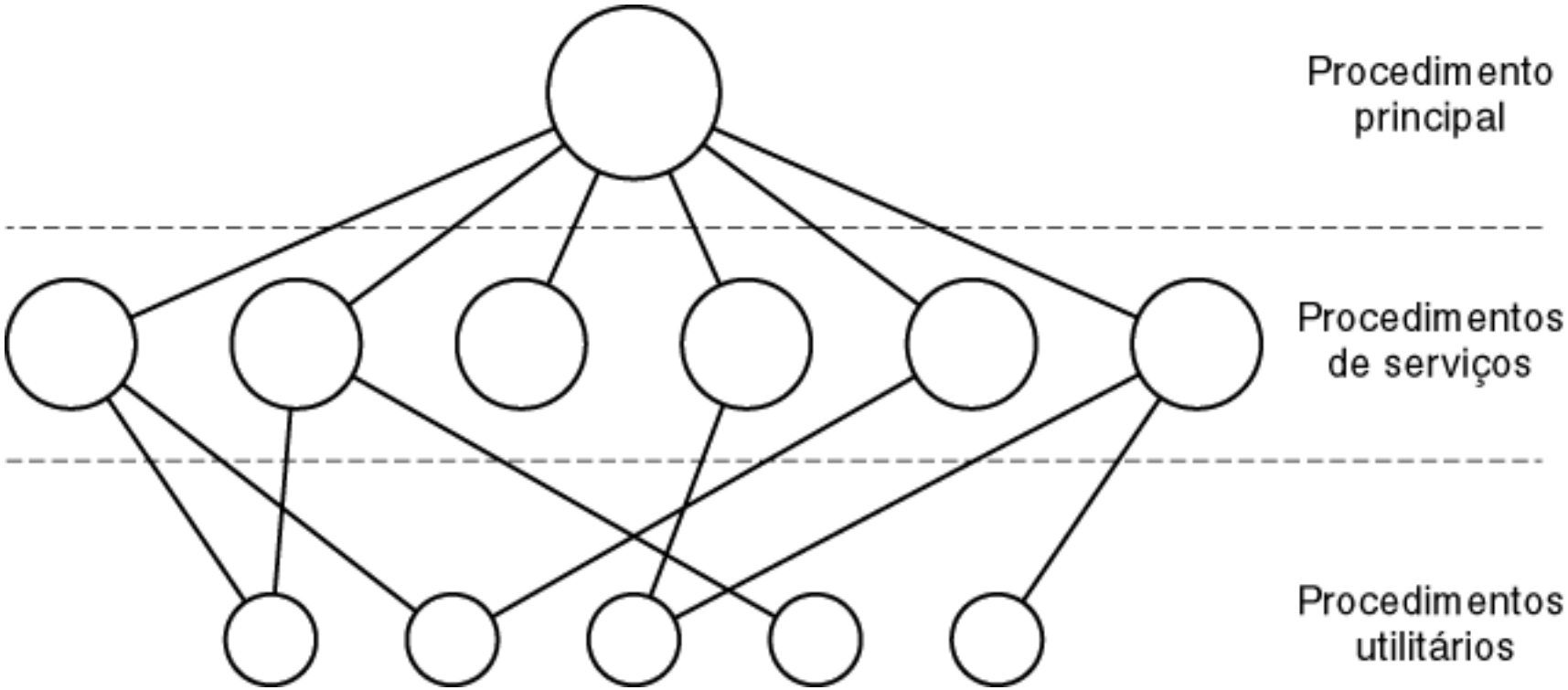
a)Sistema de arquivos antes da montagem
b)Sistema de arquivos depois da montagem

Chamadas ao Sistema (5)

Unix	Win32	Descrição
fork	CreateProcess	Crie um novo processo
waitpid	WaitForSingleObject	Pode esperar um processo sair
execve	(none)	CrieProcesso = fork + execve
exit	ExitProcess	Termine a execução
open	CreateFile	Crie um arquivo ou abra um arquivo existente
close	CloseHandle	Feche um arquivo
read	ReadFile	Leia dados de um arquivo
write	WriteFile	Escreva dados para um arquivo
lseek	SetFilePointer	Mova o ponteiro de posição do arquivo
stat	GetFileAttributesEx	Obtenha os atributos do arquivo
mkdir	CreateDirectory	Crie um novo diretório
rmdir	RemoveDirectory	Remova um diretório vazio
link	(none)	Win32 não suporta ligações (link)
unlink	DeleteFile	Destrua um arquivo existente
mount	(none)	Win32 não suporta mount
umount	(none)	Win32 não suporta mount
chdir	SetCurrentDirectory	Altere o diretório de trabalho atual
chmod	(none)	Win32 não suporta segurança (embora NT suporte)
kill	(none)	Win32 não suporta sinais
time	GetLocalTime	Obtenha o horário atual

Algumas chamadas da interface API Win32

Estrutura de Sistemas Operacionais (1)



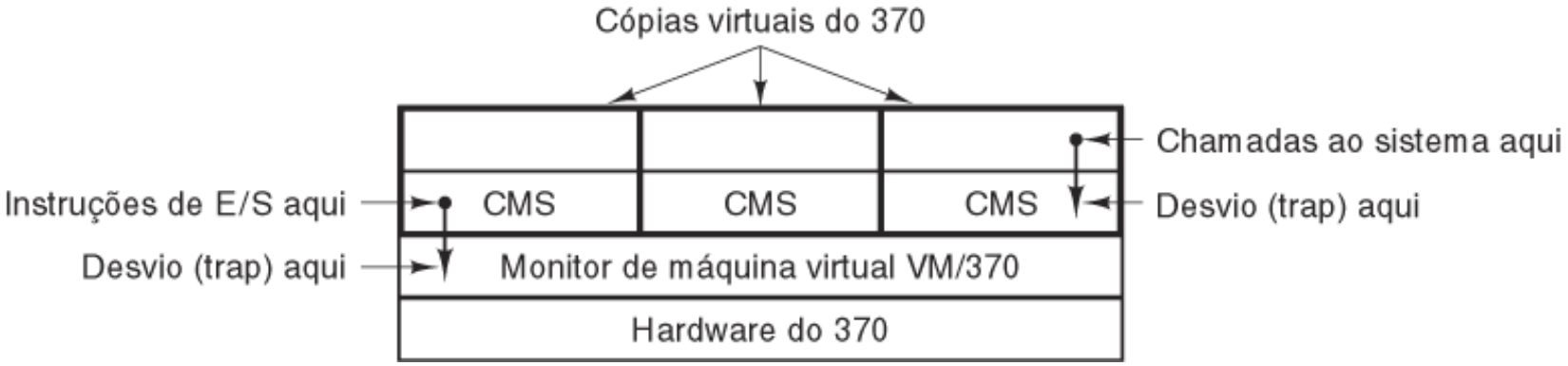
Modelo simples de estruturação de um sistema monolítico

Estrutura de Sistemas Operacionais (2)

Camada	Função
5	O operador
4	Programas do usuário
3	Gerenciamento de entrada/saída
2	Comunicação operador-processo
1	Gerenciamento da memória e do tambor magnético
0	Alocação de processador e multiprogramação

Estrutura do sistema operacional **THE**

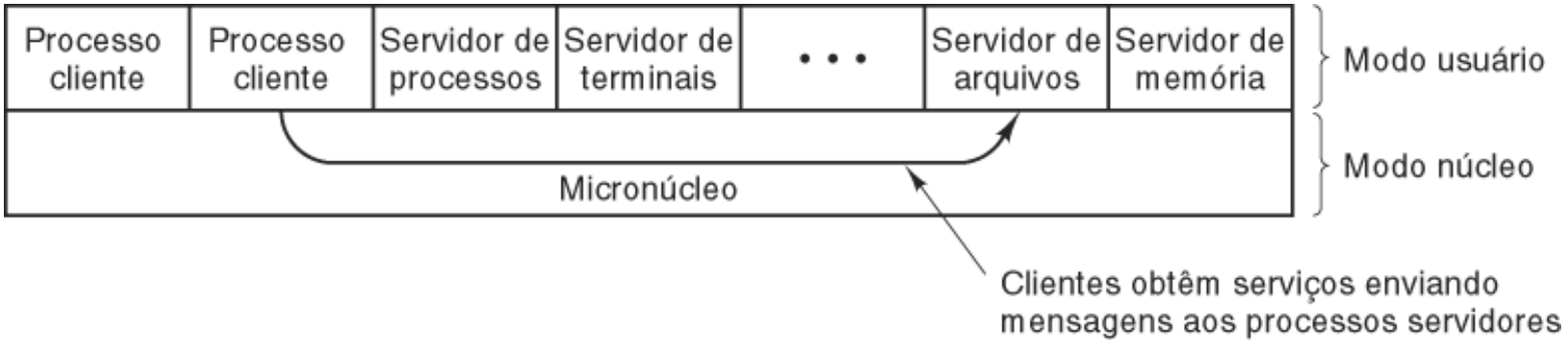
Estrutura de Sistemas Operacionais (3)



Estrutura do VM/370 (lançado em 1972); no exemplo, com instâncias do CMS (*Conversational Monitor System*)) rodando como *guest*

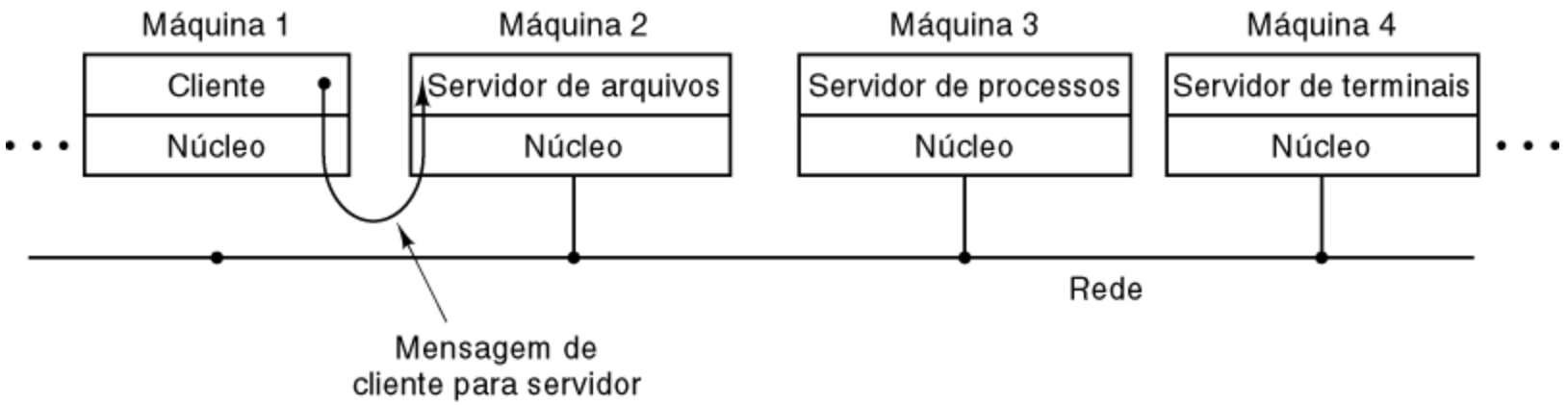
Versão atual do sistema: *zVM*

Estrutura de Sistemas Operacionais (4)



O modelo cliente-servidor

Estrutura de Sistemas Operacionais (5)



O modelo cliente-servidor em um sistema distribuído

Unidades Métricas

Exp.	Explícito	Prefixo	Exp.	Explícito	Prefixo
10^{-3}	0,001	mili	10^{3}	1 000	quilo
10^{-6}	0,000001	micro	10^{6}	1 000 000	mega
10^{-9}	0,000000001	nano	10^{9}	1 000 000 000	giga
10^{-12}	0,000000000001	pico	10^{12}	1 000 000 000 000	tera
10^{-15}	0,000000000000001	femto	10^{15}	1 000 000 000 000 000	peta
10^{-18}	0,000000000000000001	atto	10^{18}	1 000 000 000 000 000 000	exa
10^{-21}	0,000000000000000000001	zepto	10^{21}	1 000 000 000 000 000 000 000	zetta
10^{-24}	0,000000000000000000000001	yocto	10^{24}	1 000 000 000 000 000 000 000 000	yotta

Os prefixos métricos

Just to have fun playing with older Oss!!

http://www.pcjs.org/