Sistemas Operacionais

Aula 01 - Introdução

Edirlei Soares de Lima
<edirlei@iprj.uerj.br>

O que é um sistema operacional?

- Um computador moderno consiste em:
  - Um ou mais processadores;
  - Memória principal;
  - Discos;
  - Impressoras;
  - Diversos dispositivos de entrada e saída;

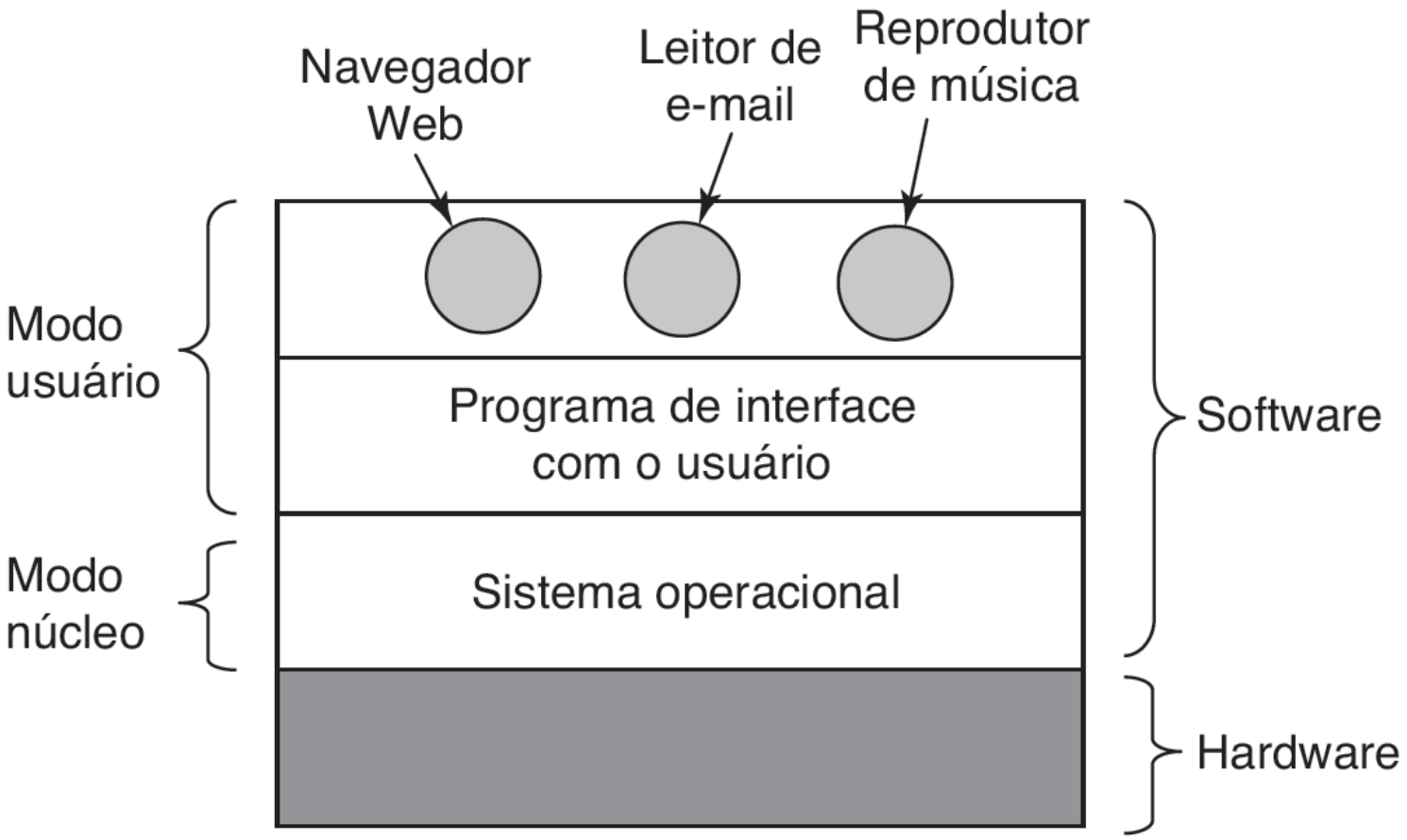
- Para gerenciar todos esses componentes é necessária uma camada de software: **o sistema operacional**.

O que é um sistema operacional?

- Escrever programas que controlam todos os componentes de harware é extremamente dificil!
  - Solução: **sistema operacional**

- **Sistema operacional:** software que controla os recursos do sistema computacional e oferece ao usuário uma interface para interagir com cada um destes recursos.

O que é um sistema operacional?



O que é um sistema operacional?

- Sistemas operacionais diferem de programas de usuário (aplicações).

- São grandes, complexos e tem vida longa.
  - Linux: mais de 5 milhões de linhas de código;

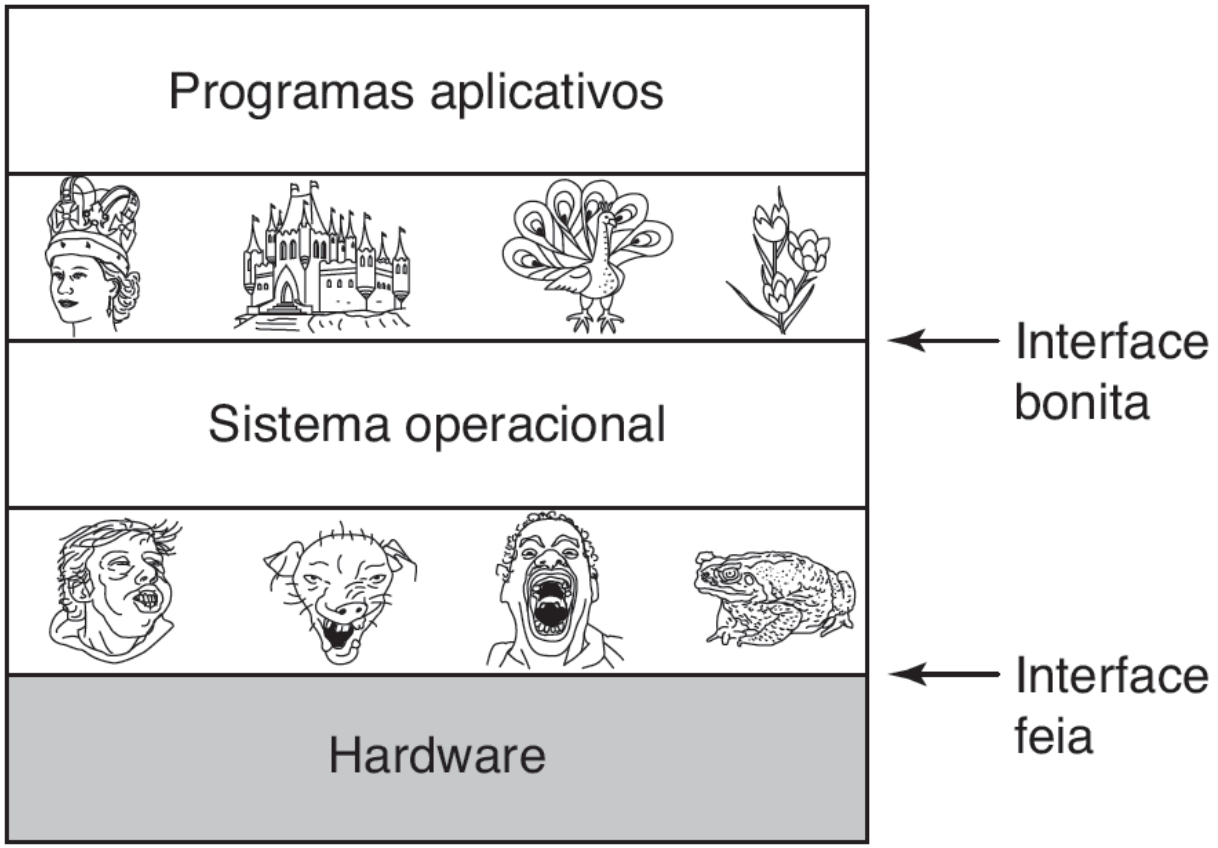
- **Importante:** o shell não é parte do sistema operacional!

O que é um sistema operacional?

- **É uma máquina estendida:**
  - Esconde uma infinidade de detalhes de operação do hardware e dispositivos periféricos;
  - Apresenta ao programador uma máquina virtual, oferecendo abstrações bem mais fáceis de serem usadas.

- **É um gerenciador de recurso:**
  - Cada programa recebe uma fatia de tempo de uso de um recurso;
  - Cada programa recebe uma fatia de espaço em um recurso.

O que é um sistema operacional?



História dos Sistemas Operacionais

- **Meados do século XIX**: Charles Babbage (1792-1871), por volta de 1833, projetou o primeiro computador. No entanto, a pouca tecnologia da época não permitiu que o projeto tivesse sucesso.

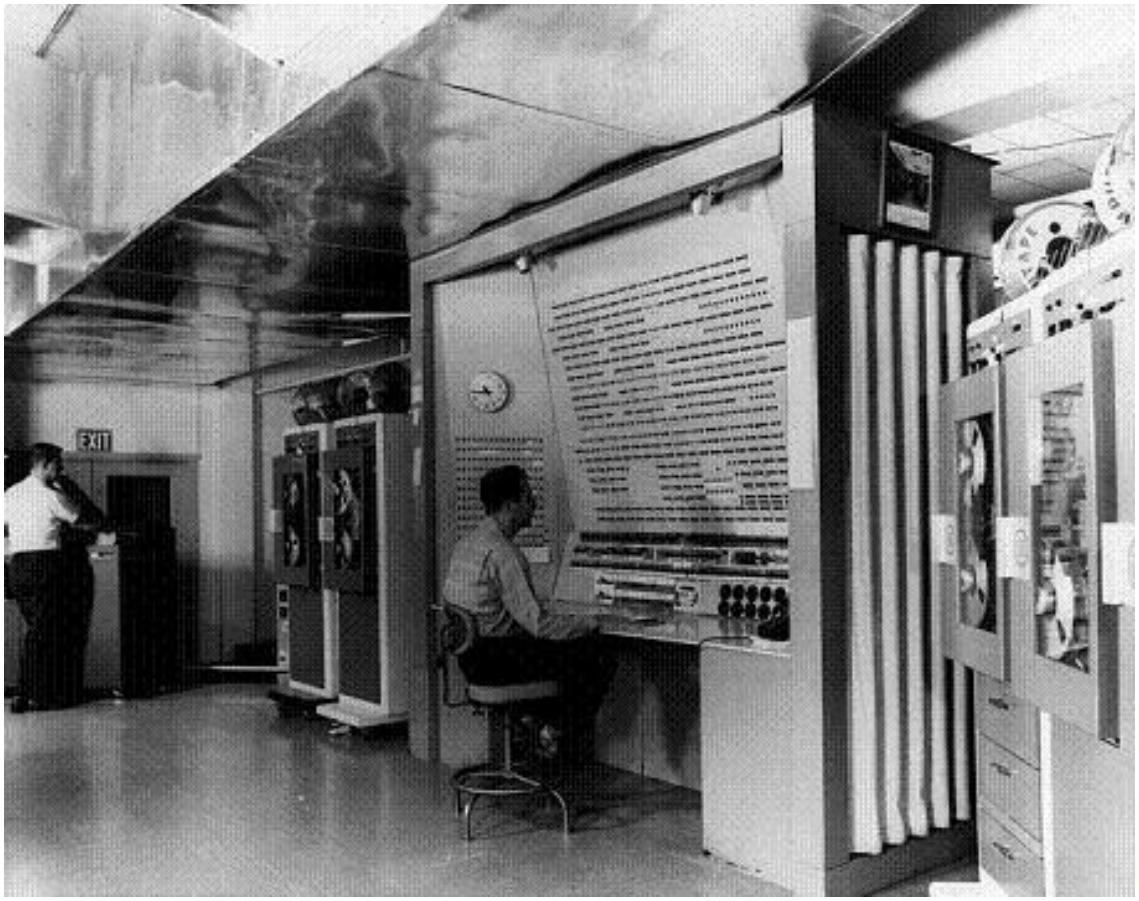
- Máquina analítica:
  - Não tinha um SO;
  - Percebeu que precisava de um software que possibilitasse seu uso;

História dos Sistemas Operacionais

- **Primeira Geração (1940-1955):** Válvulas
  - 1940: John von Neumann cria o primeiro computador digital (baseado em válvulas);
  - Máquinas enormes que ocupavam salas imensas e possuiam milhares de válvulas;
  - Não existiam ainda os conceitos de sistema operacional e linguagem de programação de alto nível;

História dos Sistemas Operacionais

- **Primeira Geração (1940-1955):** Válvulas



História dos Sistemas Operacionais

- **Primeira Geração (1940-1955):** Válvulas
  - O acesso às máquinas era feito por meio de reserva de tempo.
  - Os programadores faziam sua programação diretamente nos painéis das máquinas ("hardwired");
  - Praticamente todos os problemas submetidos eram cálculos numéricos diretos (tabelas de seno, logaritmos, etc);

História dos Sistemas Operacionais

- **Final dos anos 40:** primeiro computador eletrônico (transistores)
  - ENIAC (Electronic Numerical Integrator And Computer);

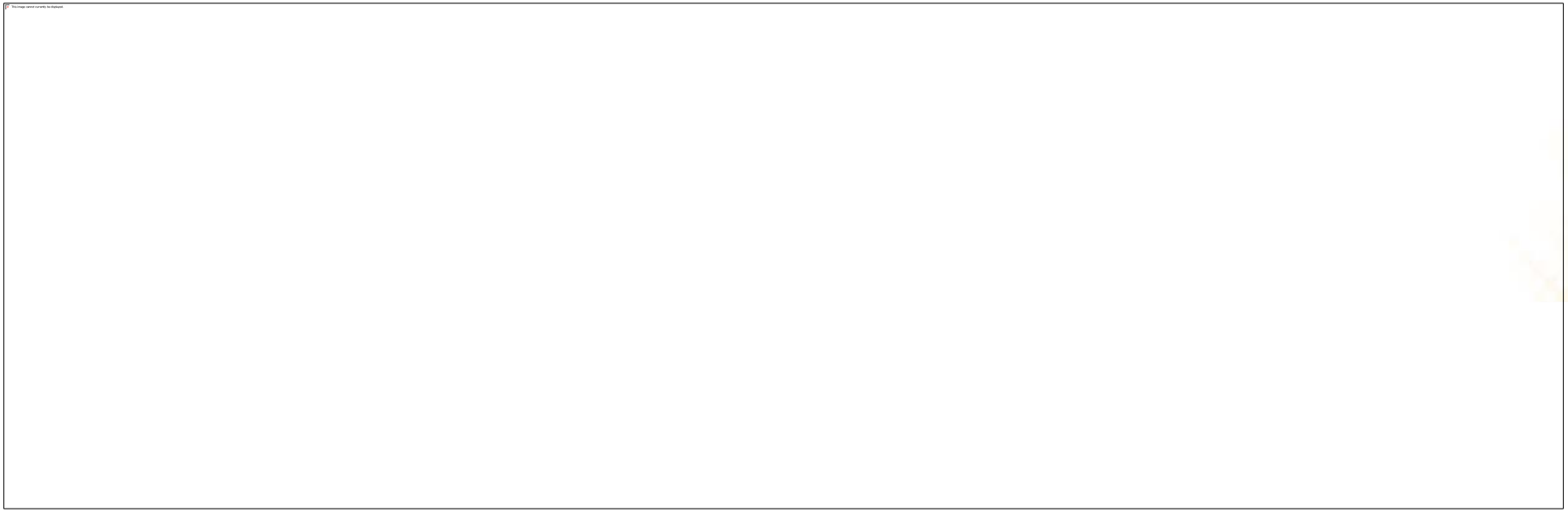
- **1950:** surgem os cartões perfurados
  - Os programas eram codificados nos cartões e sua leitura era feita por máquina;
  - John von Neumann propõe uma programação não "hardwired“: nasce o Assembler/Assembly;

História dos Sistemas Operacionais

- **Segunda Geração (1955-1965):** Transistores e Sistemas em Batch
  - O desenvolvimento dos transistores tornou o computador mais confiável possibilitando sua comercialização: mainframes;
  - No entanto, devido aos altos custos, poucos tinham acesso a essa tecnologia – somente grandes empresas, órgãos governamentais ou universidades;

História dos Sistemas Operacionais

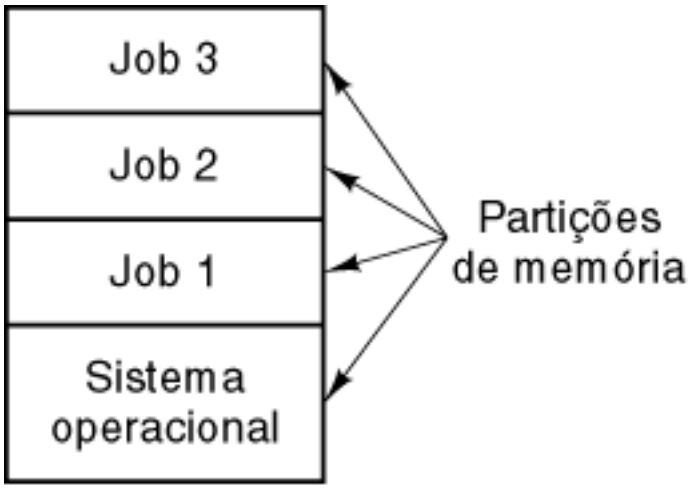
- **Segunda Geração (1955-1965):** Transistores e Sistemas em Batch
  - Cartões perfurados ainda são utilizados:



FMS (*Fortran Monitor System*)

História dos Sistemas Operacionais

- **Terceira Geração (1965-1980):** Circuitos integrados e Multiprogramação
  - System/360 (IBM): sistema operacional OS/360
    - Enorme, complexo, ineficiente, cheio de erros...
  - Multiprogramação: Dividir a memória em diversas partes e alocar a cada uma dessas partes um job.



História dos Sistemas Operacionais

- **Terceira Geração (1965-1980):** Circuitos integrados e Multiprogramação
  - Time-sharing: cada usuário tinha um terminal on-line à disposição;
    - A CPU é compartilhada e alocada para os jobs que precisam de processamento;
    - Exemplo: se 20 usuários estão ativos e 17 estão ausentes, o processador é alocado a cada um dos 3 jobs que estão sendo executados;

História dos Sistemas Operacionais

- **Quarta Geração (1980-1990):** Computadores Pessoais
  - Com a tecnologia de circuitos integrados de larga escala (LSI) surgem chips com milhares de transistores encapsulados em um centímetro quadrado de silício
    - Intel – 8080 (1974)
    - IBM– PC (início dos anos 80)
    - Apple – Apple e Macintosh

História dos Sistemas Operacionais

- **Quarta Geração (1980-1990):** Computadores Pessoais
  - Sistemas Operacionais:
    - MS-DOS, Windows...
    - UNIX, LINUX...
    - MAC OS...

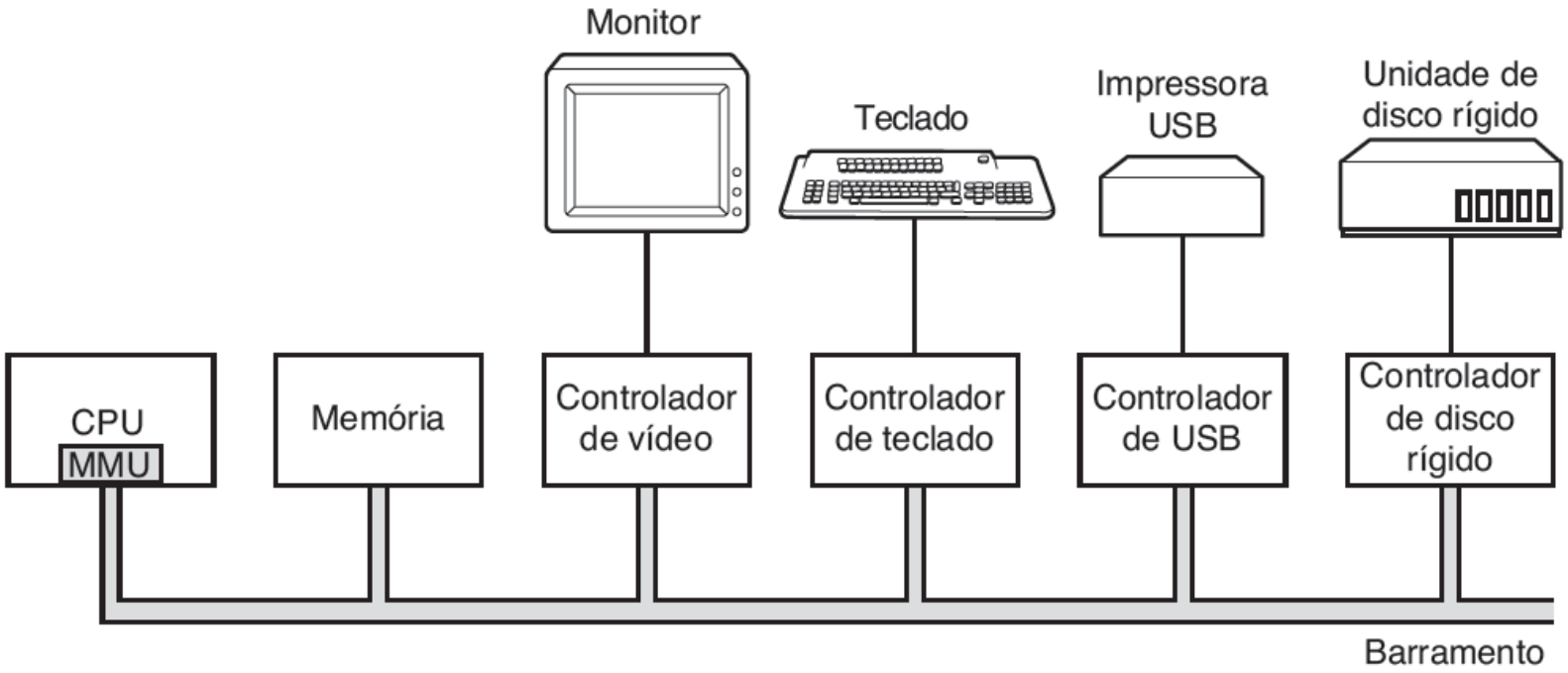
História dos Sistemas Operacionais

- **Quinta Geração (1990-hoje)**
  - Sistemas Operacionais Distribuídos:
    - Apresenta-se como um sistema operacional centralizado, mas que, na realidade, tem suas funções executadas por um conjunto de máquinas independentes;
  - Sistema Operacionais para dispositivos móveis;
    - Execução de tarefas com economia de energia (baterias limitadas);

O Zoológico de Sistemas Operacionais

- Sistemas operacionais de computadores de grande porte;
- Sistemas operacionais de servidores;
- Sistemas operacionais de computadores pessoais;
- Sistemas operacionais de multiprocessadores;
- Sistemas operacionais de tempo real;
- Sistemas operacionais embarcados;
- Sistemas operacionais para dispositivos móveis;

Revisão de Hardware



Conceitos Sobre Sistemas Operacionais

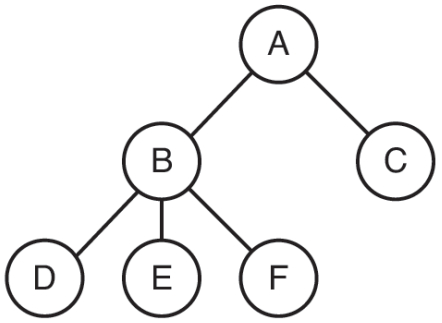
- Processos;
- Espaço de endereçamento;
- Arquivos;
- Entrada e saída;
- Segurança;

Processos

- Um processo consite em um programa em execução.
  - Está associado a um espaço de endereçamento e recursos.

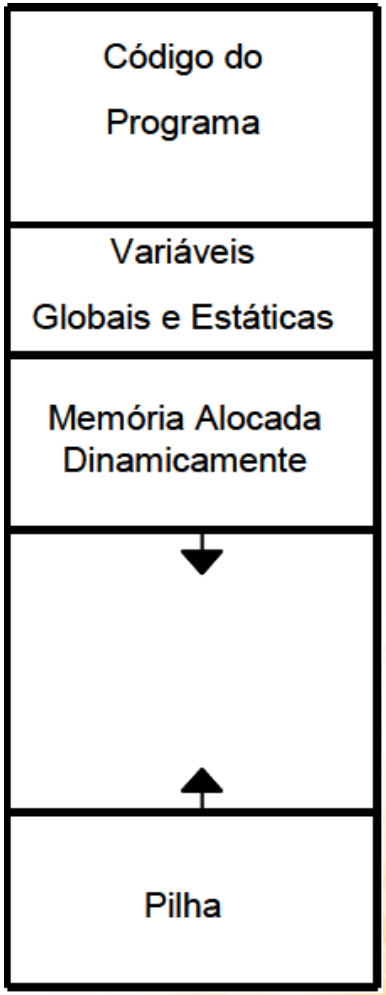
- O processo é um envelope que armazena todas as informações necessárias para a execução de um programa.

- O SO é responsável pelas seguintes atividades com relação a gerência de processos:
  - Criação e deleção de Processos.
  - Suspensão e retomada de processos.
  - Fornecimento de mecanismos para:
    - sincronização de processo
    - comunicação de processo



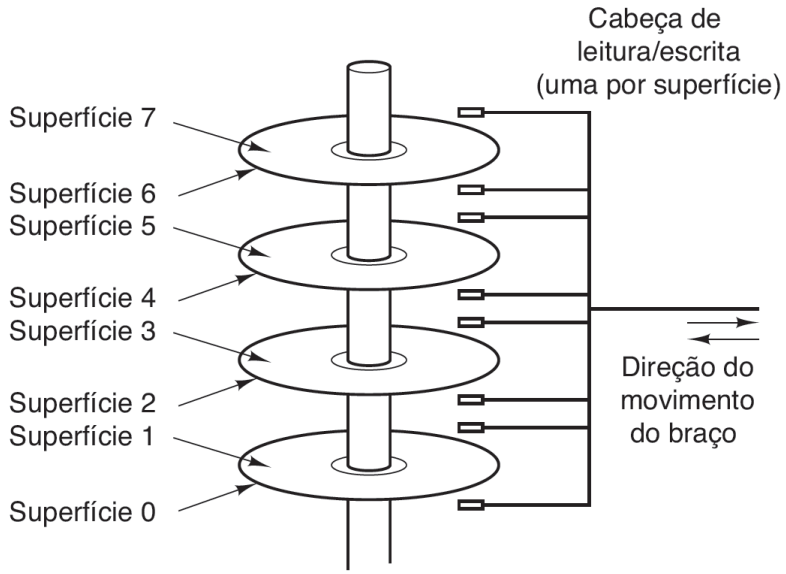
Espaço de endereçamento

- O SO é responsável pelas seguintes atividades com relação a gerência de memória (espaço de endereçamento):
  - Manter informações de que partes da memória estão em uso e por quem;
  - Decidir que processos carregar quando espaços de memória estão disponíveis;
  - Alocar e liberar espaço de memória quando necessário;

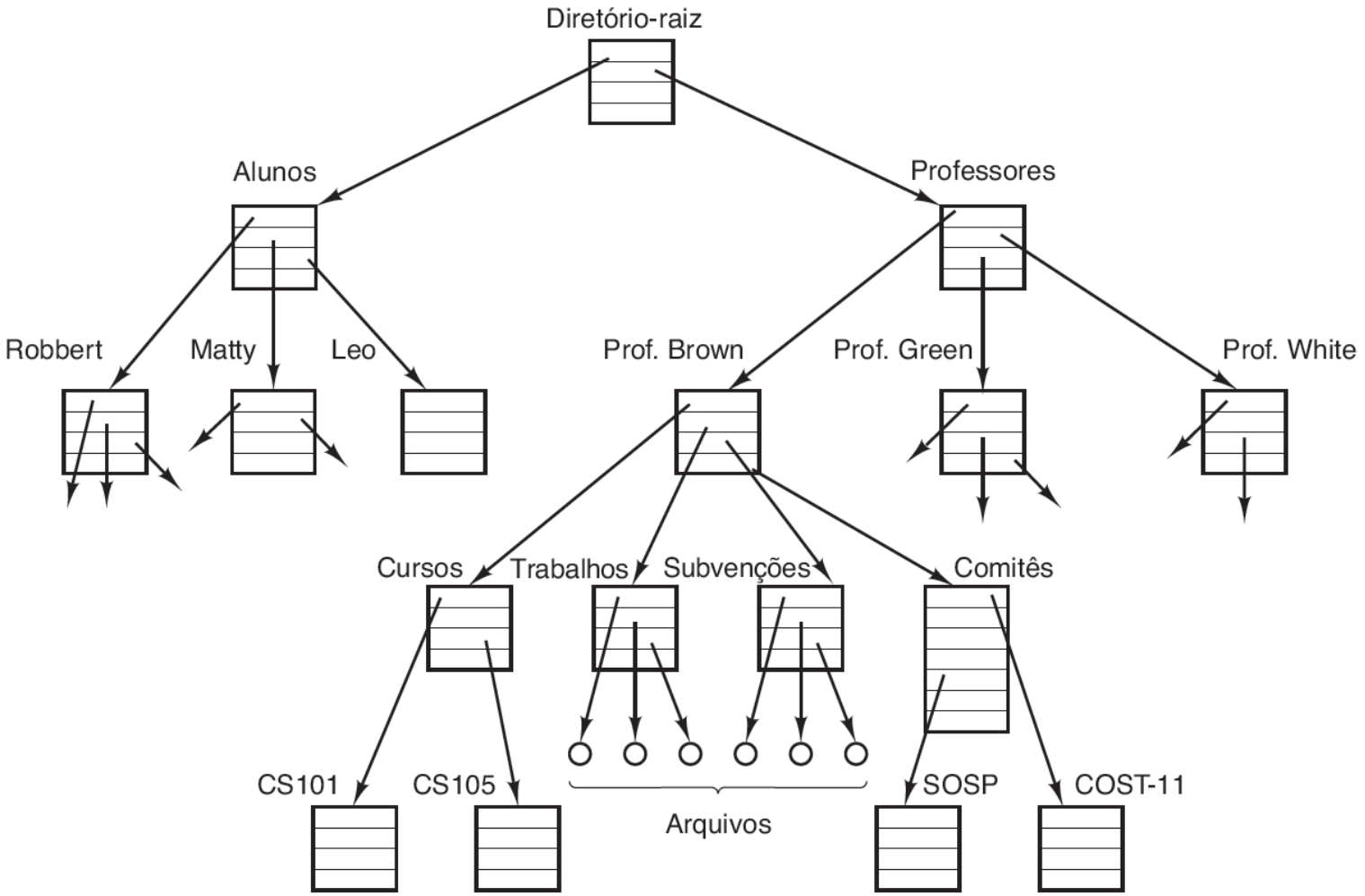


Arquivos

- O SO é responsável pelas seguintes atividades com relação a gerência de arquivos:
  - Criação e deleção de arquivo;
  - Criação e deleção de diretório;
  - Suporte de primitivas para manipular arquivos e diretórios;
  - Mapeamento de arquivos na memória secundária;



Arquivos


Diretório-raiz
Alunos
Professores
Robbert
Matty
Leo
Prof. Brown
Prof. Green
Prof. White
Cursos
Trabalhos
Subvenções
Comitês
CS101
CS105
Arquivos
SOSP
COST-11

Entrada e Saída

- Existem vários tipos de dispositivos de Entrada/Saída: teclados, monitores e impressoras .

- O SO é responsável pelo gerenciamento desses dispositivos:
  - Possui um subsistema de E/S para gerenciar seus dispositivos;
  - Alguns dos componentes de E/S são independentes de dispositivo, aplicam-se para todos os dispositivos;
  - Outros são específicos de cada dispositivos (ou família), como drivers;

Segurança

- Computadores contem muitas informações que os usuários, muitas vezes, querem manter confidenciais.

- O SO é responsável por gerenciar o sistema de segurança, para que arquivos ou outras informações somente seja acessíveis por usuários autorizados.

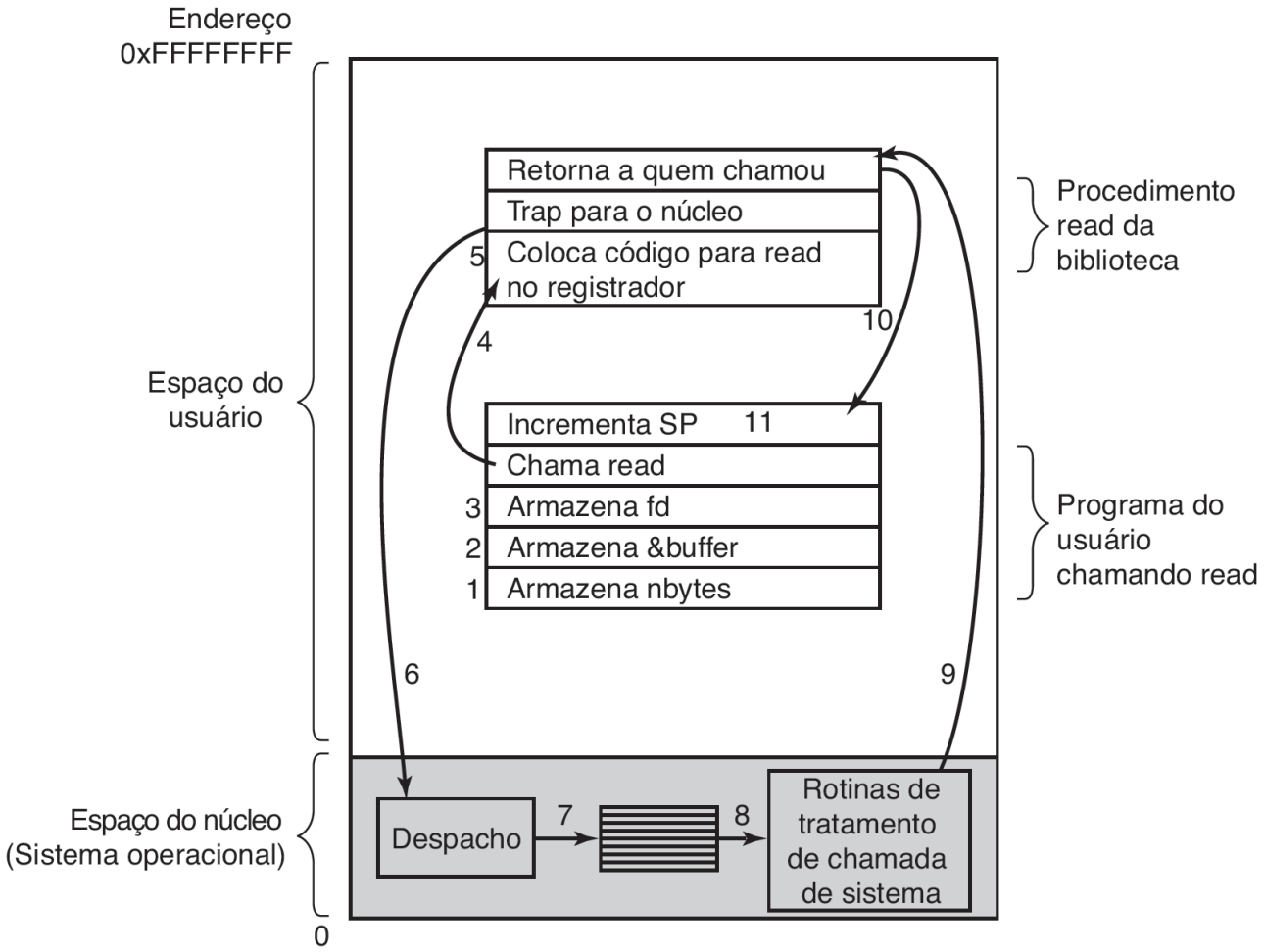
- Exemplo: permissões rwx para proteção de arquivos em UNIX:
  - Exemplo: rwxr-x—x
  - r = read access;
  - x = execute access;
  - w = write access;

Chamadas de Sistema

- Chamadas de sistema são a interface entre o SO (kernel) e os programas de usuário.
  - São funções oferecidas pelo kernel para programas de usuário.
  - Funcionam da mesma forma que funções oferecidas por bibliotecas de usuário.

- Exemplo: `int read(int fd, char *buffer, int nbytes);`
  - read – nome da chamada de sistema;
  - fd – descritor do arquivo;
  - buffer – nome do local de armazenamento ;
  - nbytes – numero de bytes a ler.

Chamadas de Sistema



Chamadas de Sistema

Gerenciamento de processos

Chamada	Descrição
pid = fork( )	Cria um processo filho idêntico ao pai
pid = waitpid(pid, &statloc, options)	Espera que um processo filho seja concluído
s = execve(name, argv, environp)	Substitui a imagem do núcleo de um processo
exit(status)	Conclui a execução do processo e devolve status

Gerenciamento de arquivos

Chamada	Descrição
Fd = open(file, how, ...)	Abre um arquivo para leitura, escrita ou ambos
s = close(fd)	Fecha um arquivo aberto
n = read(fd, buffer, nbytes)	Lê dados a partir de um arquivo em um buffer
n = write(fd, buffer, nbytes)	Escreve dados a partir de um buffer em um arquivo
position = lseek(fd, offset, whence)	Move o ponteiro do arquivo
s = stat(name, &buf)	Obtém informações sobre um arquivo

Chamadas de Sistema

Gerenciamento do sistema de diretório e arquivo

Chamada	Descrição
s = mkdir(name, mode)	Cria um novo diretório
s = rmdir(name)	Remove um diretório vazio
s = link(name1, name2)	Cria uma nova entrada, name2, apontando para name1
s = unlink(name)	Remove uma entrada de diretório
s = mount(special, name, flag)	Monta um sistema de arquivos
s = umount(special)	Desmonta um sistema de arquivos

Chamadas de Sistema

UNIX	Win32	Descrição
fork	CreateProcess	Cria um novo processo
waitpid	WaitForSingleObject	Pode esperar que um processo saia
execve	(nenhuma)	CreateProcess = fork + execve
exit	ExitProcess	Conclui a execução
open	CreateFile	Cria um arquivo ou abre um arquivo existente
close	CloseHandle	Fecha um arquivo
read	ReadFile	Lê dados a partir de um arquivo
write	WriteFile	Escreve dados em um arquivo
lseek	SetFilePointer	Move o ponteiro do arquivo

Chamadas de Sistema

stat	GetFileAttributesEx	Obtém vários atributos do arquivo
mkdir	CreateDirectory	Cria um novo diretório
rmdir	RemoveDirectory	Remove um diretório vazio
link	(nenhuma)	Win32 não dá suporte a links
unlink	DeleteFile	Destrói um arquivo existente
mount	(nenhuma)	Win32 não dá suporte a mount
umount	(nenhuma)	Win32 não dá suporte a mount
chdir	SetCurrentDirectory	Altera o diretório de trabalho atual
chmod	(nenhuma)	Win32 não dá suporte a segurança (embora o NT suporte)
kill	(nenhuma)	Win32 não dá suporte a sinais
time	GetLocalTime	Obtém o tempo atual

Exercícios

Lista de Exercícios 01

http://www.inf.puc-rio.br/~elima/so/

Leitura Complementar

- Andrew S. Tanenbaum. **Sistemas Operacionais Modernos**, 3ª Edição, Pearson, 2010.

- **Capitulo 1: Introdução**