Processos e Threads

Prof. Dr. José Luís Zem
Prof. Dr. Renato Kraide Soffner
Prof. Ms. Rossano Pablo Pinto

Faculdade de Tecnologia de Americana
Centro Paula Souza

Tópicos

- Conceito de Processos
- Escalonamento de Processos
- Operações em Processos
- Processos Cooperantes
- Comunicação entre Processos
- Comunicação em Sistemas Cliente-Servidor
- Introdução aos Threads
- Modelos de Multithreading

Conceito de Processos

- Um sistema operacional executa uma variedade de programas:
  - Sistemas Batch – jobs,
  - Sistemas de Tempo Compartilhado - programas de usuários e tarefas.
- Processo pode ser, inicialmente, definido como um programa em execução.
  - Execução de processo ocorre de maneira seqüencial
  - apenas uma instrução por vez é executada*
- Um processo possui (informações sobre seu estado atual):
  - Registradores (incluindo contador de instruções), Pilhas e Filas, Seção de dados, Seção de instruções.

Conceito de Processos

- 2+ processos podem estar associados a um mesmo programa
  - Mas: possuem seqüências de instruções distintas
- É comum ter um processo que crie vários processos durante sua execução
  - VERIFICAR SAÍDA DO pstree:
    - [] número de processos
    - [{}] número de threads

Conceito de Processos

```
init─┬─NetworkManager
     ├─acpid
     ├─atd
     ├─avahi-daemon───avahi-daemon
     ├─avahi-dnsconfd
     ├─chipcardd4───chipcardd4
     ├─console-kit-dae───63*[{console-kit-dae}]
     ├─cron
     ├─cupsd
     ├─3*[dbus-daemon]
     ├─2*[dbus-launch]
     ├─dd
     ├─dhcdbd
     ├─dhclient3
     ├─freshclam
     ├─gconfd-2
     ├─6*[getty]
     ├─guidance-power-
     ├─hald───hald-runner─┬─hald-addon-acpi
     │                    ├─hald-addon-cpuf
     │                    ├─hald-addon-inpu
     │                    └─2*[hald-addon-stor]
     ├─ica
     ├─kaccess
     ├─kded4
     ├─kdeinit4─┬─firefox-2───run-mozilla.sh───firefox-2-bin───8*[{firefox-2-bin}]
     │          ├─ica-launcher─┬─avahi-publish-s
     │          │               └─ica─┬─ica
     │          │                     └─2*[{ica}]
     │          ├─klauncher
     │          ├─ksmserver─┬─kwin
     │          │           ├─skype───5*[{skype}]
     │          │           ├─xterm───bash
     │          │           └─{ksmserver}
     │          └─2*[python]
     .....
```

Estados do Processo

- Um processo pode assumir vários estados durante o seu ciclo de vida:
  - novo: o processo está sendo criado.
  - executando: instruções estão sendo executadas.
  - bloqueado: o processo está aguardando algum evento ou resposta de uma operação de I/O.
  - pronto: o processo está aguardando para ser processado pela CPU.
  - encerrado: o processo finalizou suas execuções.

Diagrama de Estados do Processo

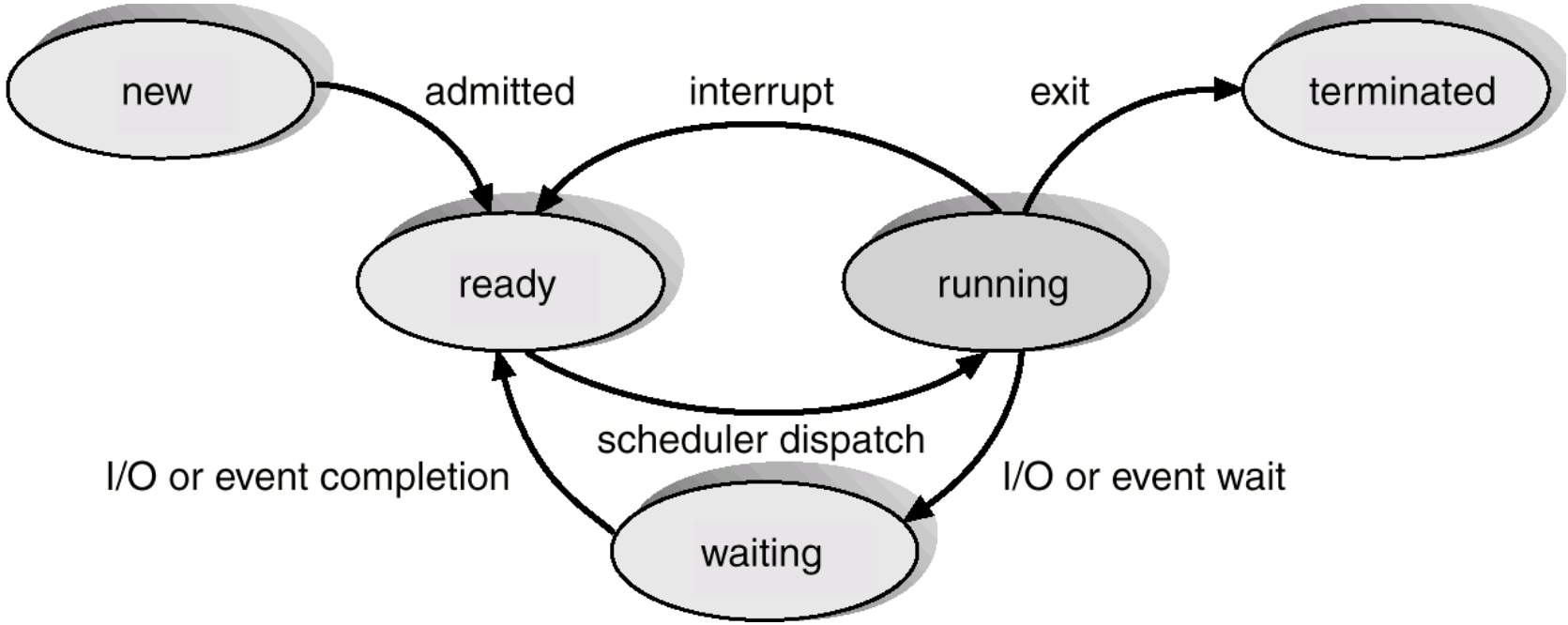


Figura 4.1

Bloco de Controle de Processos (BCP)

- Informações associadas a cada processo.
  - Estado do Processo.
  - Contador de Programas.
  - Registradores da CPU.
  - Informações sobre escalonamento.
  - Informações sobre gerenciamento de memória.
  - Informações sobre contabilidade.
  - Informações sobre estado de I/O.

Bloco de Controle de Processos (BCP)

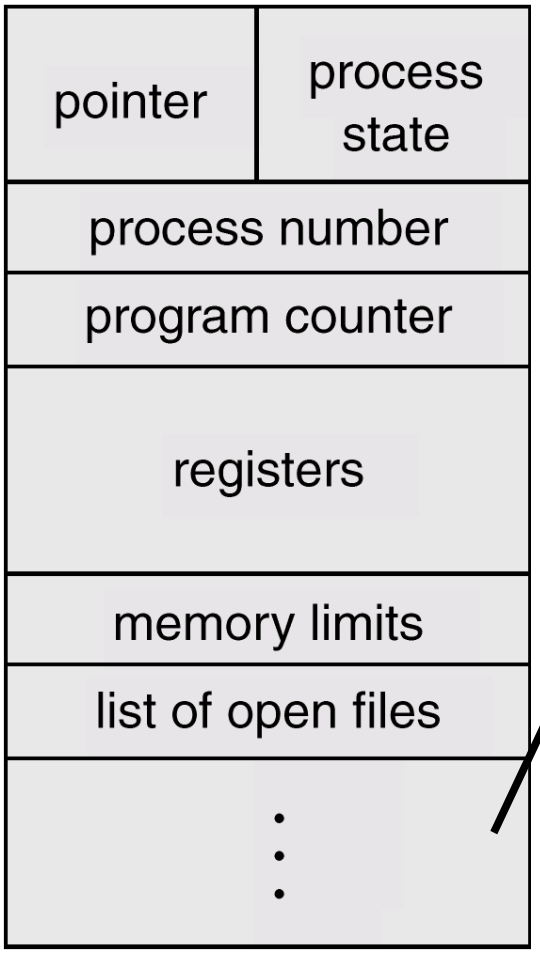


Figura 4.2

- prioridade
- contabilidade
- estado de operações de E/S

TAREFA:

1 - Explorar o /proc do Linux:
Associar os dados do PCB com as informações presentes no /proc

CPU alternando entre processos

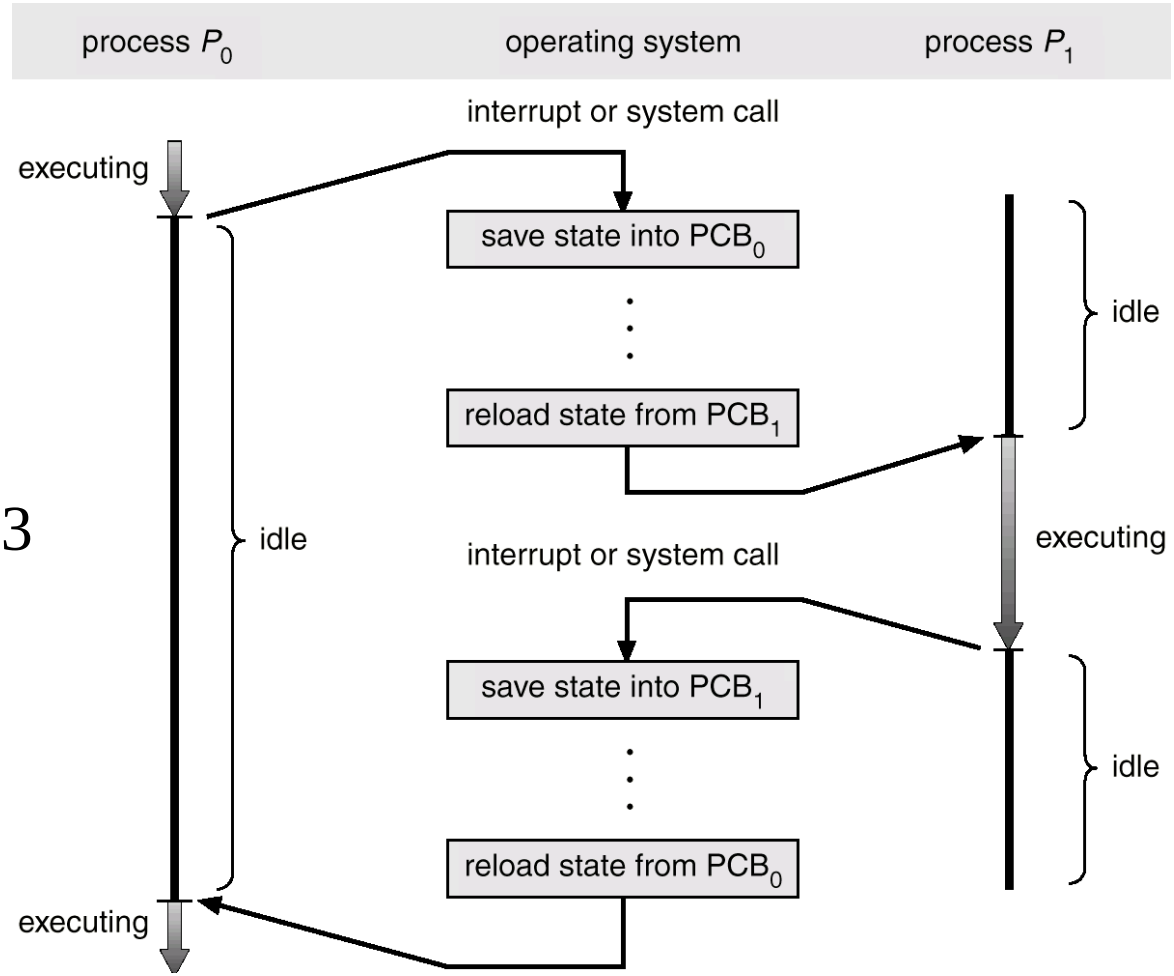


Figura 4.3

Filas de Escalonamento de Processos

- Fila de Jobs - conjunto de todos os processos do sistema.
- Fila de Prontos - conjunto de todos os processos residentes na memória principal, prontos e esperando para serem executados.
- Fila de Dispositivos - conjunto de processos aguardando por um dispositivo de I/O.
- Processos migram entre as várias filas.

Fila de Prontos e de Dispositivos de I/O

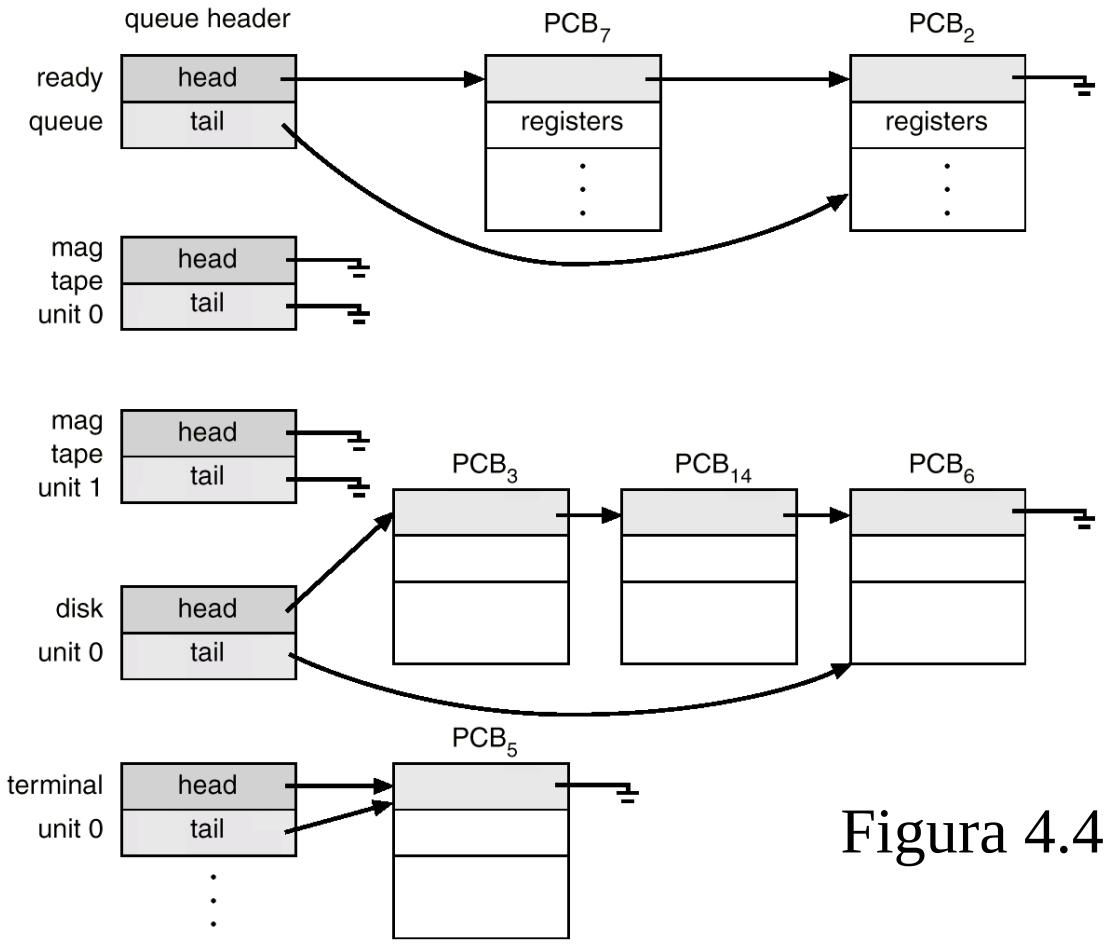


Figura 4.4

Representação do Escalonamento de Processos

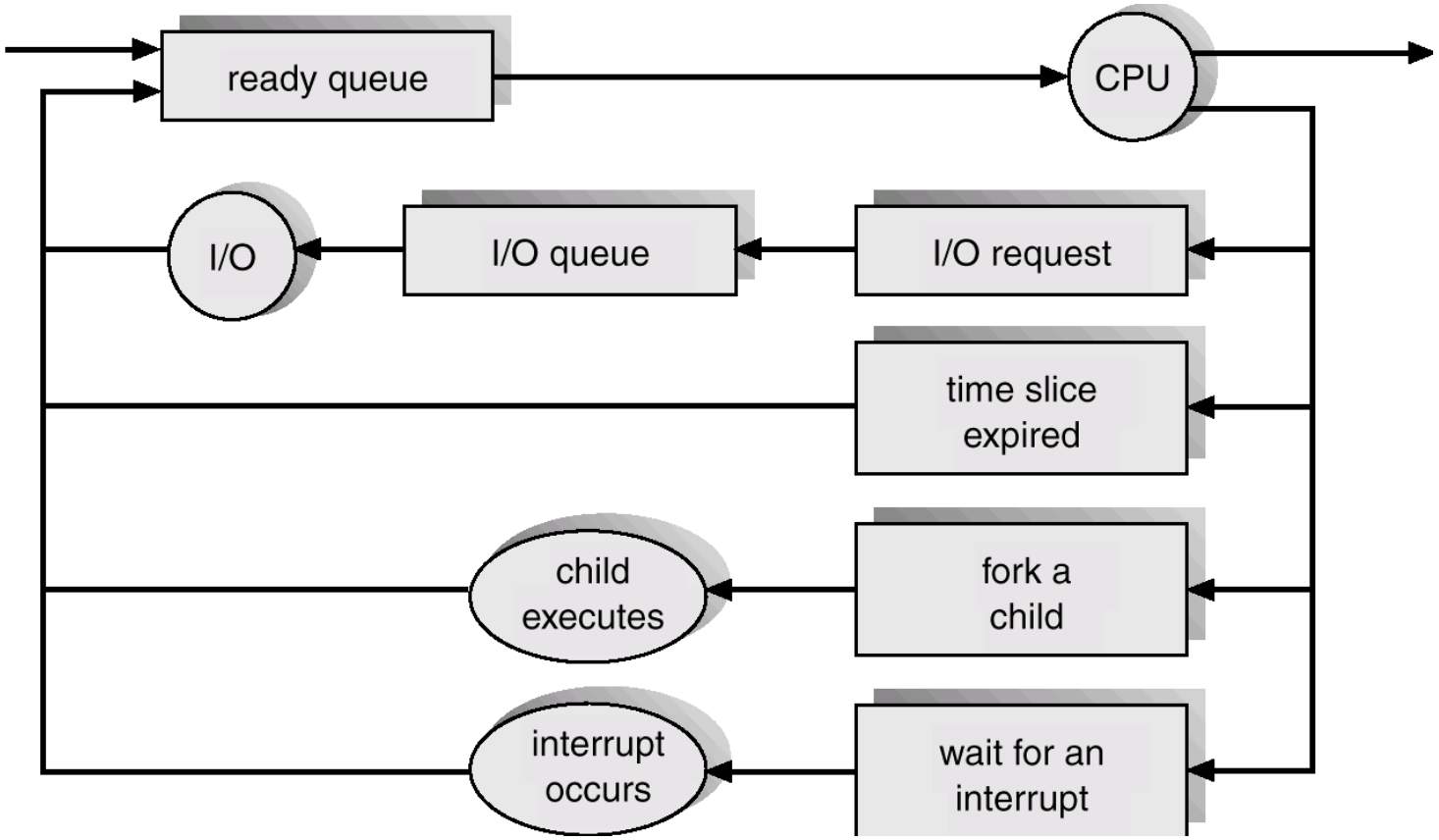


Figura 4.5

Escalonadores

- Escalonador de Longo Prazo (escalonador de jobs)
  - Seleciona quais processos deverão ser colocados na fila de prontos.
  - É solicitado quando um novo processo deve ser admitido no sistema.
  - Controla o grau de multiprogramação.
- Escalonador de Curto Prazo (escalonador de CPU)
  - Seleciona qual processo deverá ser executado após obter a CPU.
  - É constantemente solicitado.
- Escalonador de Médio Prazo

Escalonador de Médio Prazo

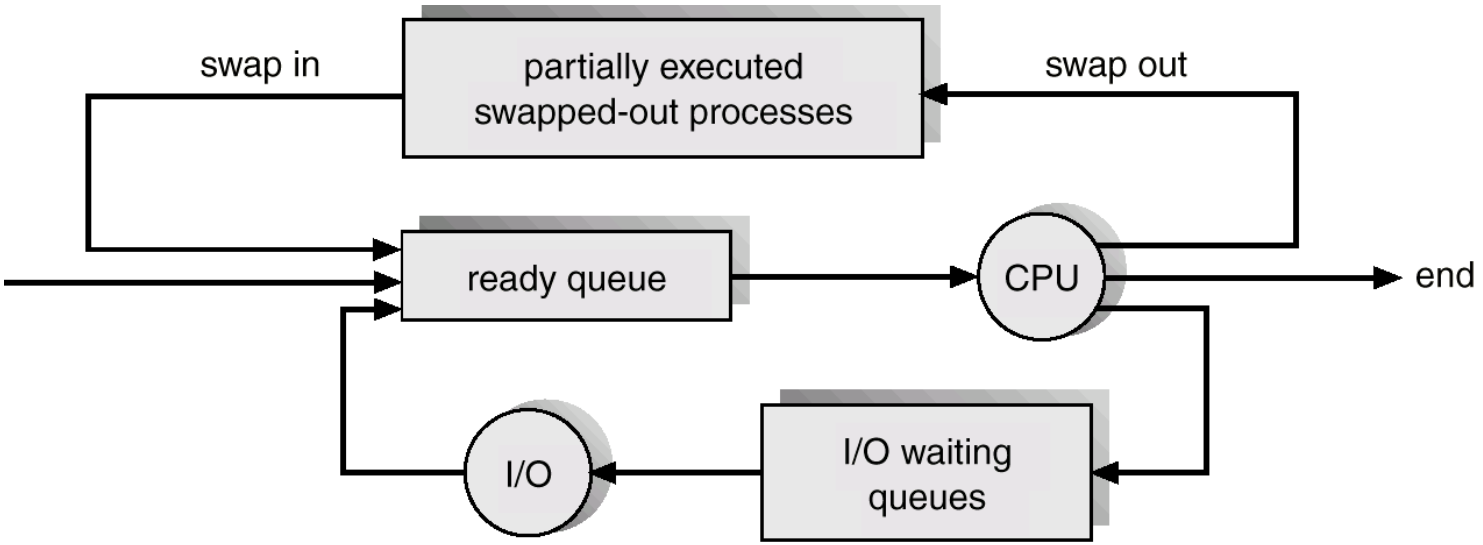


Figura 4.6

Escalonadores

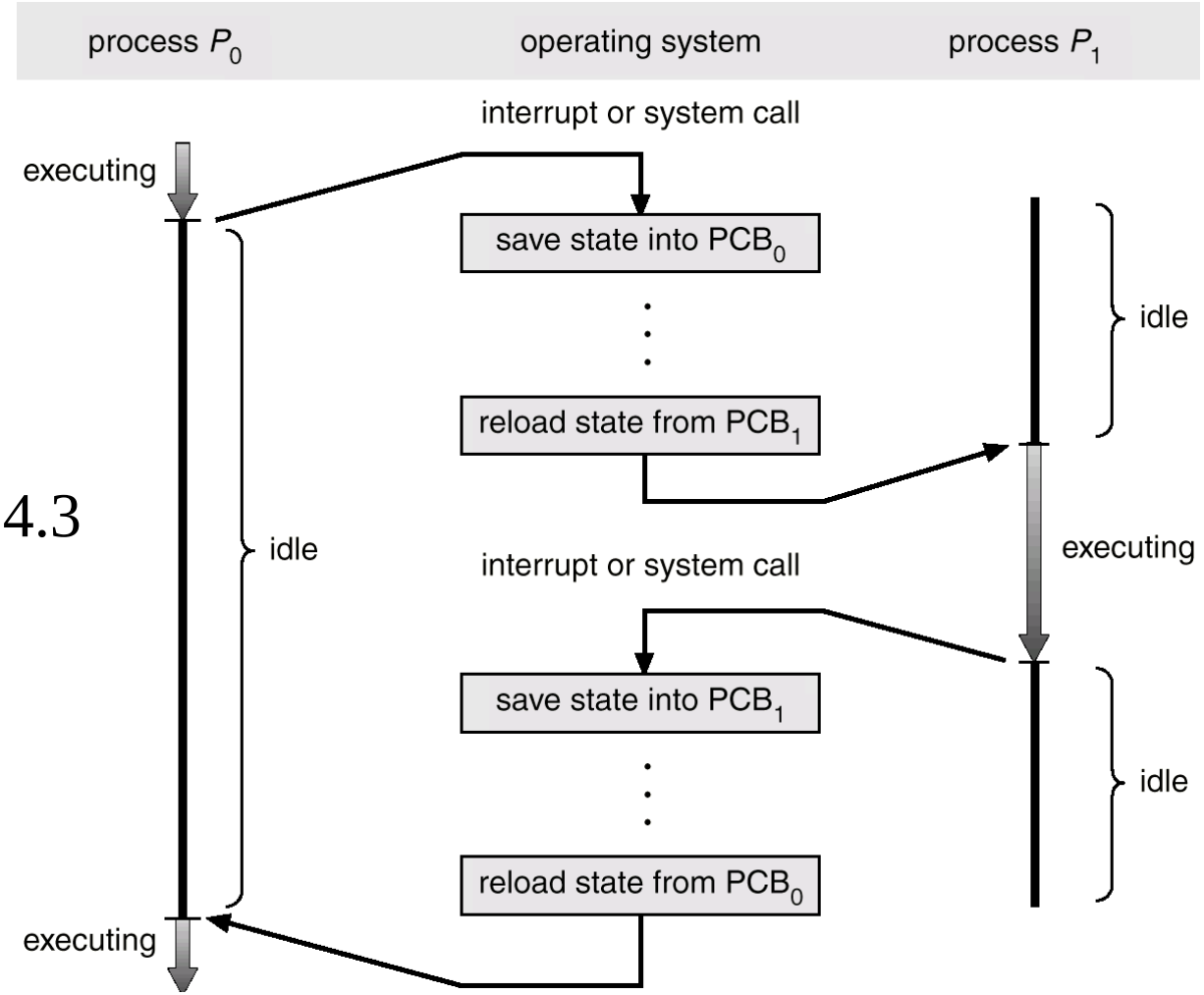
- Processos podem ser descritos como sendo:
- Processos I/O Bound
  - Gasta mais tempo realizando operações de I/O que computações, com ciclos de CPU curtos.
- Processos CPU Bound
  - Gasta mais tempo realizando computações que operações de I/O, com ciclos de CPU mais longos.

Mudança de Contexto

- Quando a CPU altera para um outro processo, o sistema deve armazenar o estado do processo antigo e carregar o estado armazenado do novo processo.
- O tempo gasto para a mudança de contexto não é útil aos processos (overhead).
- Este tempo gasto é dependente das características do hardware.

Mudança de Contexto

Figura 4.3



Processos Cooperantes

- Um Processo Independente não pode afeta ou ser afetado pela execução de um outro processo.
- Processos Cooperantes pode afetar e serem afetados pela execução de outros processos.
- Vantagens:
  - Compartilhamento de informação.
  - Speedup computacional
  - Modularidade
  - Conveniência

Comunicação entre Processos (IPC)

- Mecanismos para os processos viabilizarem a comunicação e a sincronização.
- Sistema de Mensagens - processo comunica-se com os demais sem utilizar-se de memória compartilhada.
- O IPC fornece duas operações:
  - send(message) – mensagem de tamanho fixo ou variável.
  - receive(message)
- Um processo deve estabelecer um canal de comunicação e em seguida iniciar a troca de mensagens através do send/receive.

Sincronização

- A passagem de mensagem pode ser Bloqueante e Não Bloqueante.
- A abordagem Bloqueante é considerada síncrona.
- A abordagem Não Bloqueante é considerada assíncrona.
- As primitivas send/receive podem ser bloqueantes ou não bloqueantes.

Bufferização

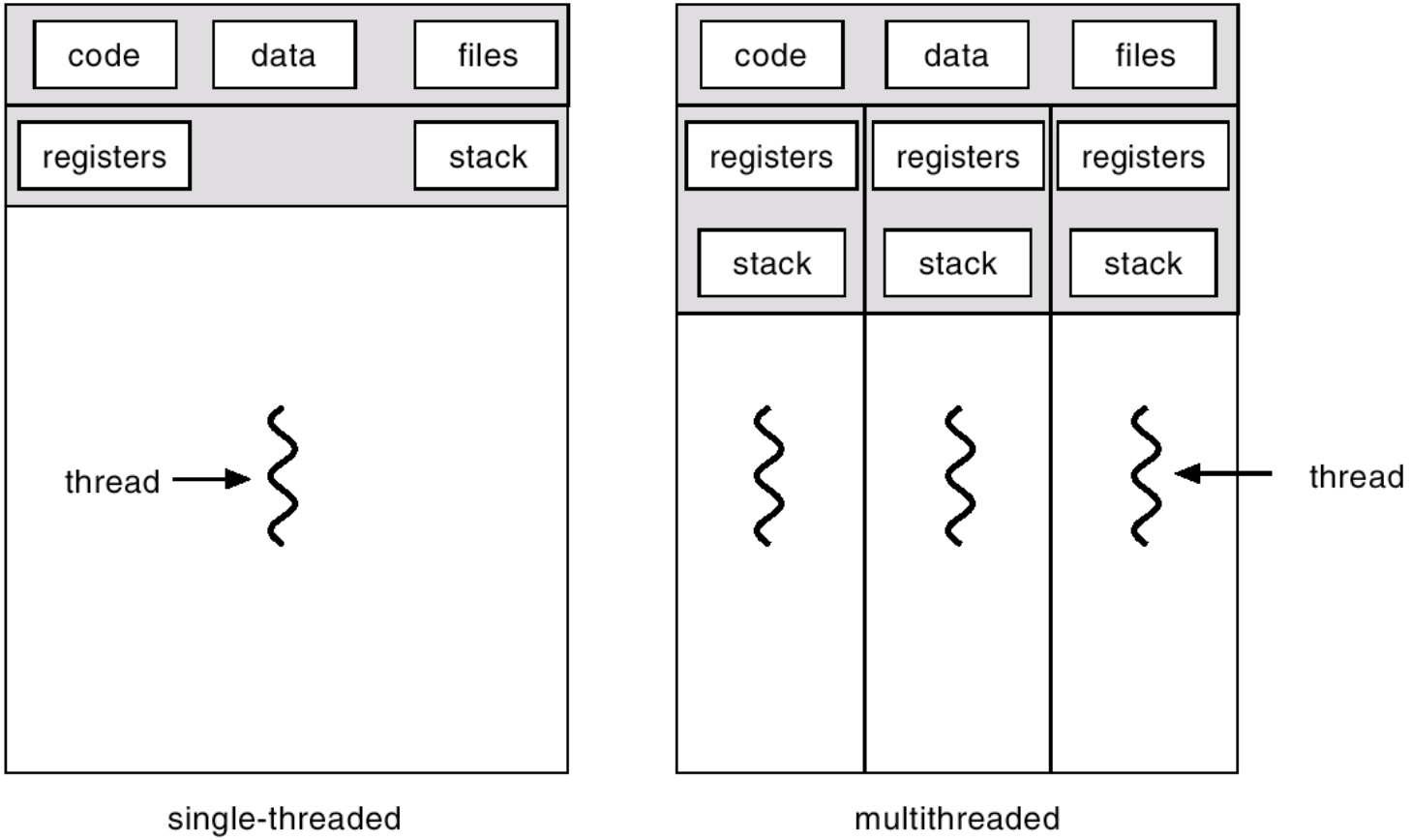
- As filas de mensagens implementam uma destas três abordagens:

- Zero Capacity:
  - sem capacidade de armazenamento
  - o emissor deve esperar pelo receptor - Rendezvous

- Bounded Capacity:
  - tamanho finito de n mensagens.
  - emissor deve esperar se o canal estiver cheio

- Unbounded Capacity:
  - tamanho infinito
  - o emissor nunca espera

Processos Pesados e Leves



Benefícios

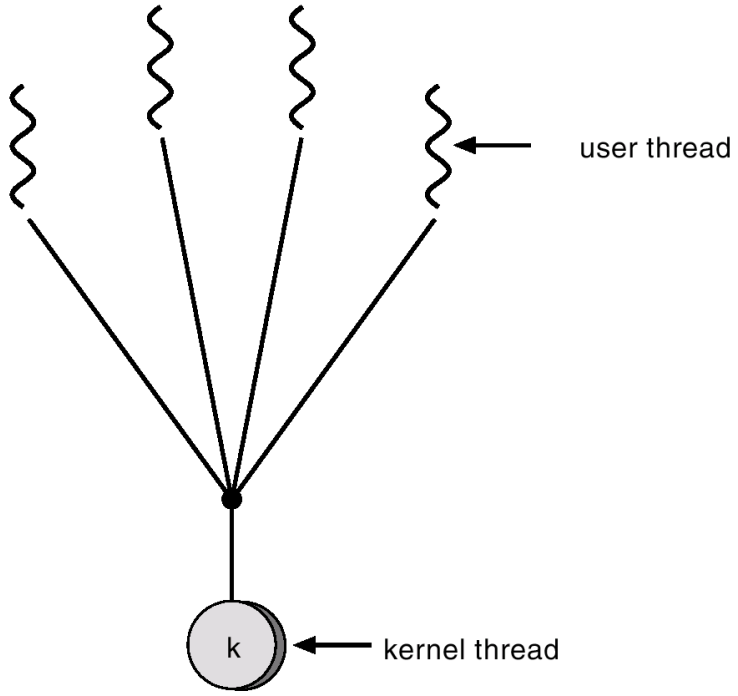
- Capacidade de Resposta
- Compartilhamento de Recursos
- Economia
- Utilização de arquiteturas multiprocessadas.

Níveis e Modelos

- Níveis de Threads
  - Thread de Nível do Usuário
    - O gerenciamento de threads é feito por uma biblioteca no nível do usuário.
  - Thread de Nível do Kernel
    - O gerenciamento é feito pelo próprio kernel do sistema operacional.
- Modelos de Multithreading
  - Muitos para Um (many-to-one)
  - Um para Um (one-to-one)
  - Muitos para Muitos (many-to-many)

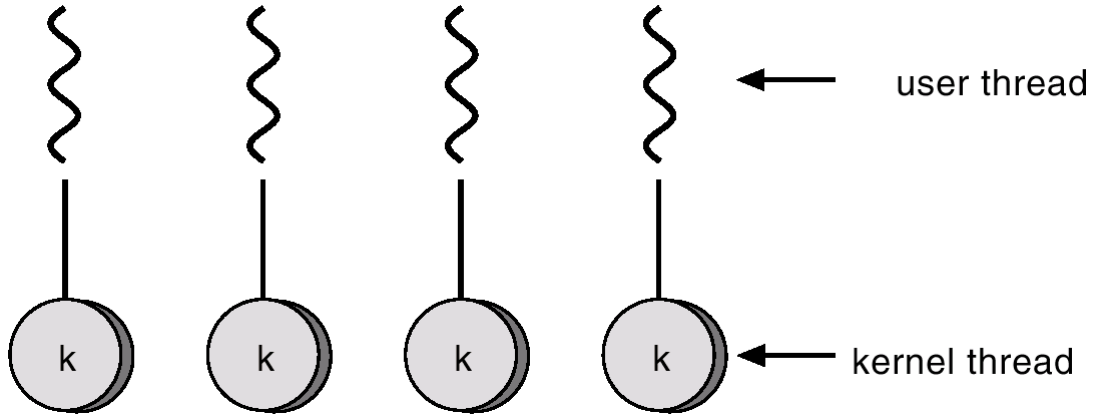
Modelo Muitos para Um

- Muitos threads do nível de usuário são mapeados em um único thread de kernel.
- Usado em sistemas que não suportam threads de kernel.



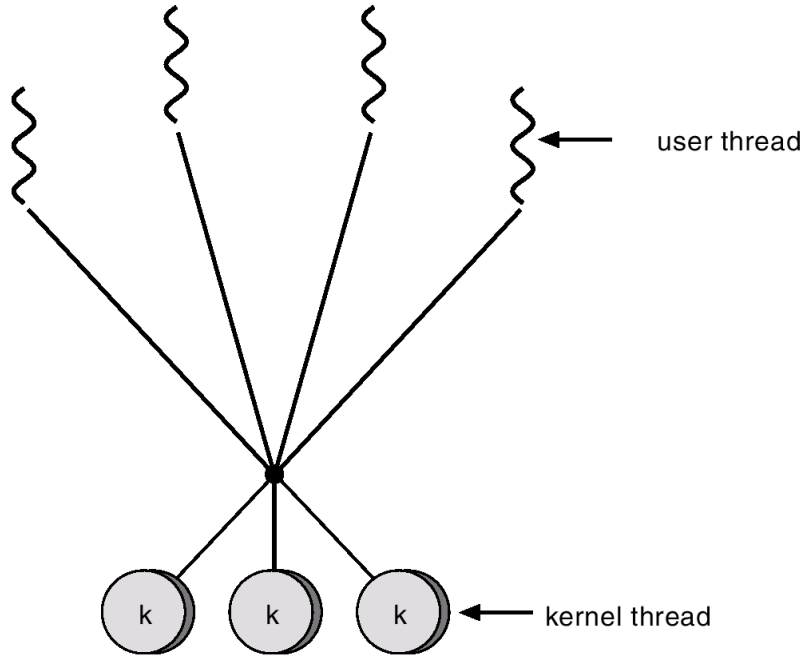
Modelo Um para Um

- Cada thread de nível do usuário é mapeado para um thread de nível do kernel.



Modelo Muitos para Muitos

- Permite que muitos threads de nível do usuário sejam mapeados para muitos threads de nível do kernel.
- Permite que o sistema operacional crie um número suficiente de threads de kernel.



Processos e Threads

Prof. Dr. José Luís Zem
Prof. Dr. Renato Kraide Soffner
Prof. Ms. Rossano Pablo Pinto

Faculdade de Tecnologia de Americana
Centro Paula Souza