Sistemas Operacionais

PROCESSOS
DESCRIÇÃO E CONTROLE

Conceito de Processo

- Um sistema operacional gerencia uma variedade de processos:
  - Sistemas batch – serviços (jobs)
  - Sistemas de compartilhamento de tempo – programas de usuários ou tarefas.
- Livros usam os termos “job” e “processo” praticamente sem distinção.
- Processo: Um programa em execução. A execução de processos segue um padrão sequencial.
- Um processo inclui:
  - contador de instruções
  - pilha
  - área de dados

Conceito de Processo

- **PROGRAMA X PROCESSO**

- **“processo é um programa em execução”**
- **o processo possui um estado e um contexto (“ é um programa com alma”)**
- **o desempenho previsto para o programa é teórico**
- **o desempenho de um processo é relativo ao ambiente que o aloja (HW, SO, outros processos)**

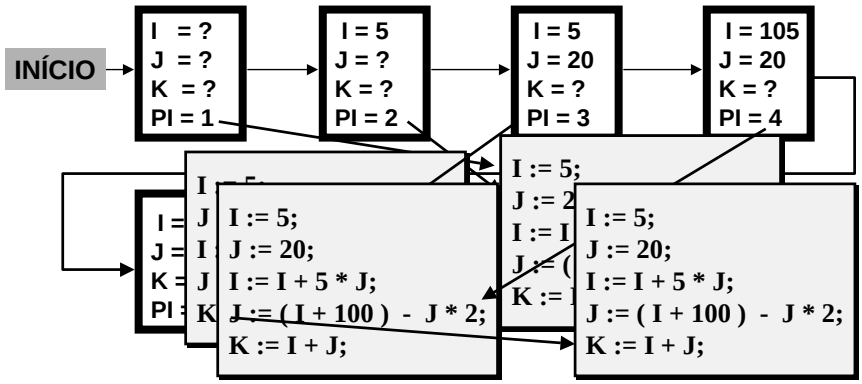
Conceito de Processo

- **PROGRAMA PASCAL PARA CALCULAR O VALOR NUMÉRICO DE UMA EXPRESSÃO**

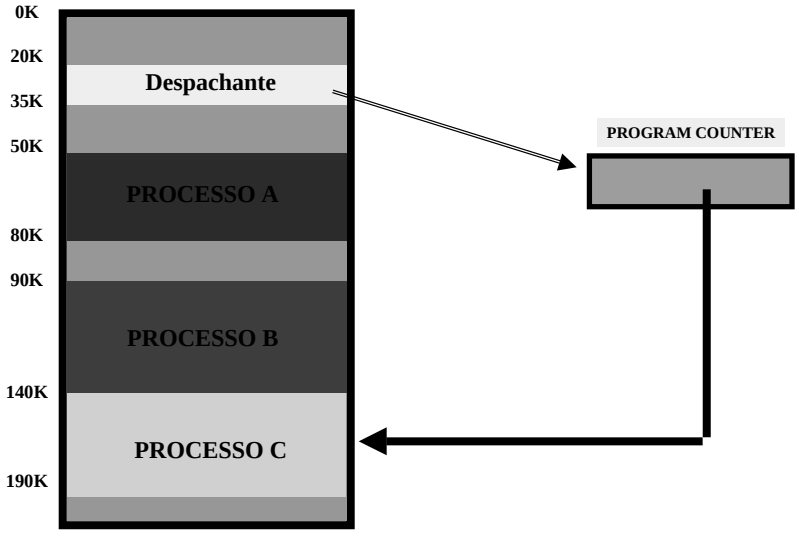
```
PROGRAM SIMPLES;
LAL 1,2,3,4,5;
VAR I,J,K : INTEGER;
BEGIN
    1:    I := 5;
    2:    J := 20;
    3:    I := I + 5 * J;
    4:    J := ( I + 100 )  -  J * 2;
    5:    K := I + J;
END.
```

Conceito de Processo

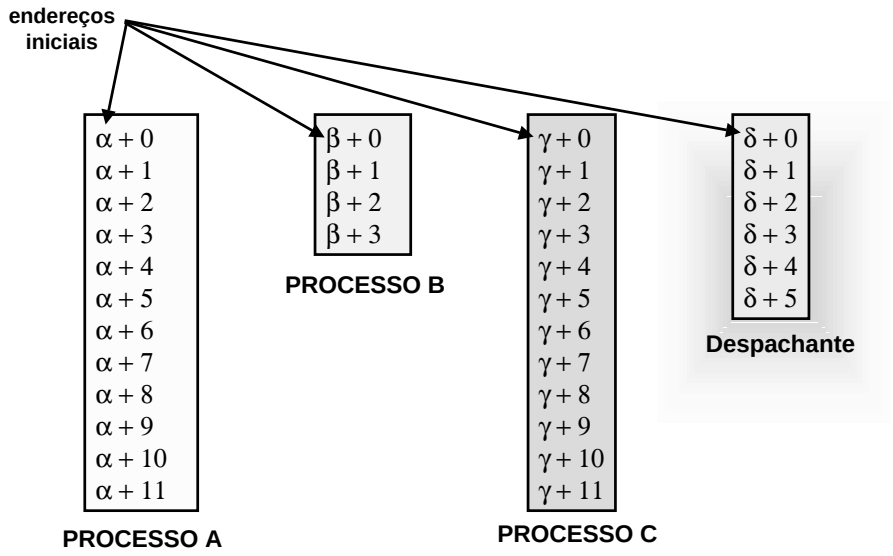
- PROGRAMA <u>SIMPLES</u> EM EXECUÇÃO: PROCESSO



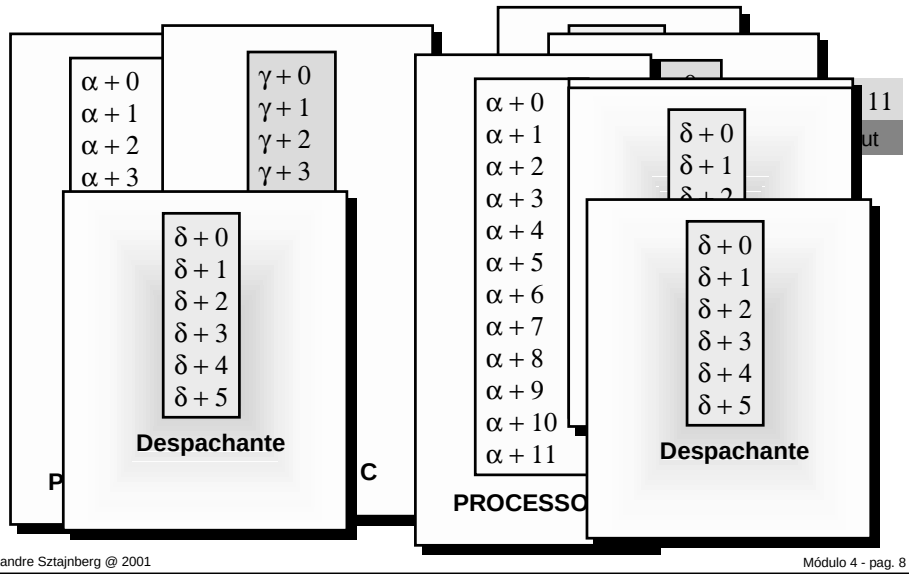
ESTADOS DO PROCESSO - EXEMPLO



TRACE DOS PROCESSOS



TRACE DA EXECUÇÃO DE A,B e C



o modelo de dois estados

Diagrama de transição

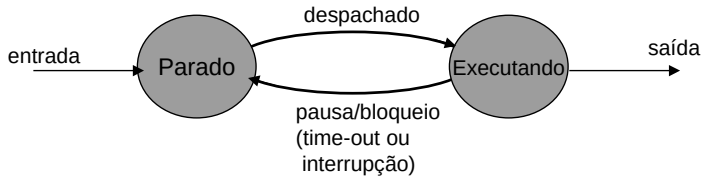
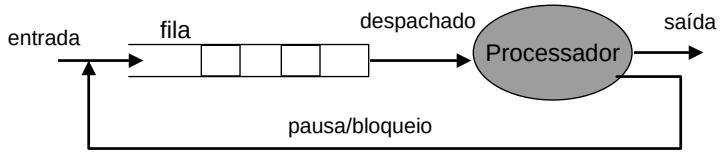
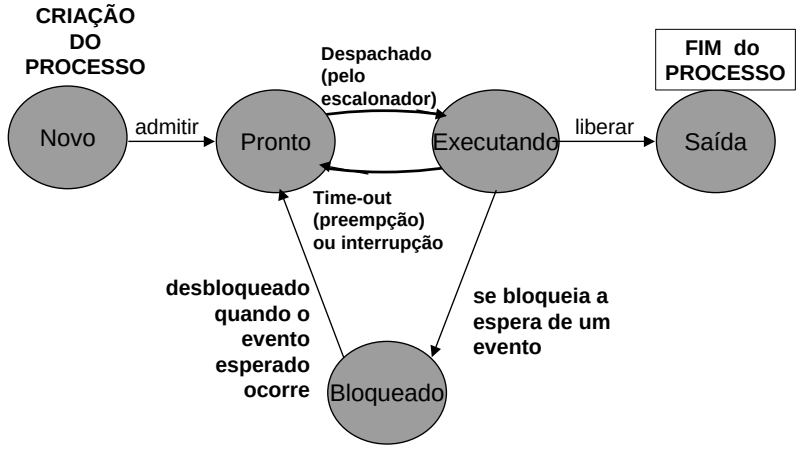


Diagrama de fila



O MODELO DOS CINCO ESTADOS



Estados de um Processo

- **Como um processo executa, ele muda de *estado***
  - **novo: O processo está sendo criado.**
  - **executando: Suas instruções estão sendo executadas.**
  - **espera: O processo está esperando pela ocorrência de algum evento.**
  - **pronto: O processo está esperando para ser escalado para a execução.**
  - **terminado: O processo terminou sua execução.**

TRANSIÇÕES possíveis ENTRE OS ESTADOS

- **NENHUM → NOVO**
- **NOVO → PRONTO**
- **PRONTO → EXECUTANDO**
- **EXECUTANDO → TERMINADO**
- **EXECUTANDO → PRONTO**
- **EXECUTANDO → BLOQUEADO**
- **BLOQUEADO → PRONTO**
- **PRONTO → TERMINADO**
- **BLOQUEADO → TERMINADO**

ESTADOS DOS PROCESSOS NO TRACE

TEMPO	PROCESSO A	PROCESSO B	PROCESSO C
1-6	EXECUTANDO	PRONTO	PRONTO
7-12	PRONTO	PRONTO	PRONTO
13-18	PRONTO	EXECUTANDO	PRONTO
19-24	PRONTO	BLOQUEADO	PRONTO
25-28	PRONTO	BLOQUEADO	EXECUTANDO
29-34	PRONTO	BLOQUEADO	PRONTO
35-40	EXECUTANDO	BLOQUEADO	PRONTO
41-46	PRONTO	BLOQUEADO	PRONTO
47-52	PRONTO	BLOQUEADO	EXECUTANDO

Modelo de Filas com várias Filas de *Bloqueado*

Diagrama de fila

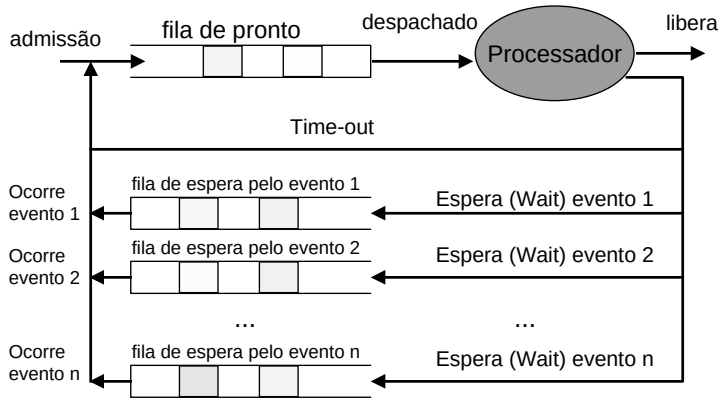
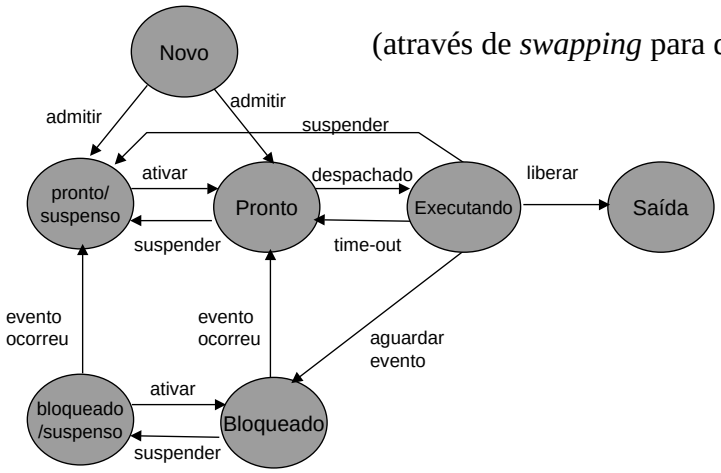


Diagrama de Transição de Estado com estado *Suspenso*



(através de *swapping* para disco)

ESTADOS DO PROCESSO

- **ESTADOS SUSPENSOS**

PRONTO : PROCESSO NA MEMÓRIA PRINCIPAL PRONTO PARA EXECUTAR

BLOQUEADO : PROCESSO NA MEMÓRIA PRINCIPAL AGUARDANDO UM EVENTO

BLOQUEADO SUSPENSO : PROCESSO NA MEMÓRIA SECUNDÁRIA AGUARDANDO UM EVENTO

PRONTO SUSPENSO : PROCESSO NA MEMÓRIA SECUNDÁRIA PRONTO PARA EXECUTAR

ESTADOS DO PROCESSO

RAZÕES PARA SUSPENDER UM PROCESSO

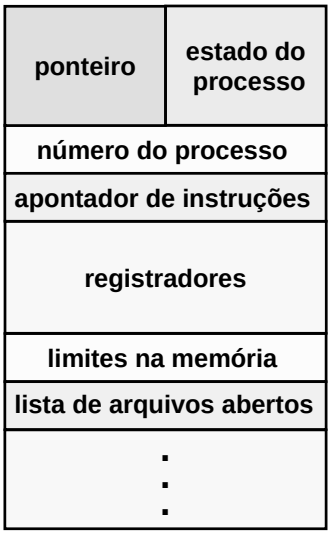
- **SWAPPING : O SO PRECISA LIBERAR ESPAÇO NA MEMÓRIA PRINCIPAL PARA TRAZER OUTRO PROCESSO DA MEMÓRIA SECUNDÁRIA.**
- **OUTRAS RAZÕES DO SO : O SO PODE SUSPENDER UM PROCESSO QUE ESTEJA RODANDO EM BACKGROUND, UM PROCESSO UTILITÁRIO, OU UM PROCESSO SUSPEITO DE ESTAR CAUSANDO PROBLEMAS.**
- **SOLICITAÇÃO DE USUÁRIO INTERATIVO**
- **TEMPORIZAÇÃO : DETERMINADOS PROCESSOS SÃO EXECUTADOS PERIODICAMENTE.**
- **SOLICITAÇÃO DO PROCESSO PAI**

Process Control Block (PCB)

Informações associadas à cada processo:

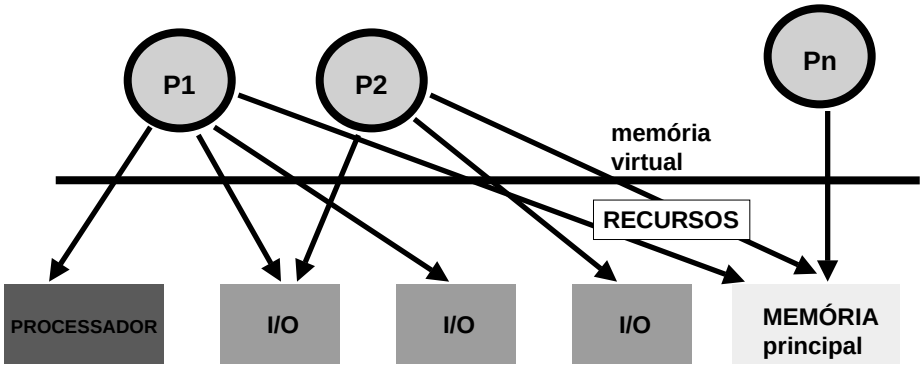
- **Estado do processo**
- **Apontador de instruções**
- **Registradores da CPU**
- **Informações relativas ao escalonamento da CPU**
- **Informações relativas ao gerenciamento de memória**
- **Dados para contabilidade (estatísticas)**
- **Informações relativas ao "status" de I/O**

Process Control Block (PCB)

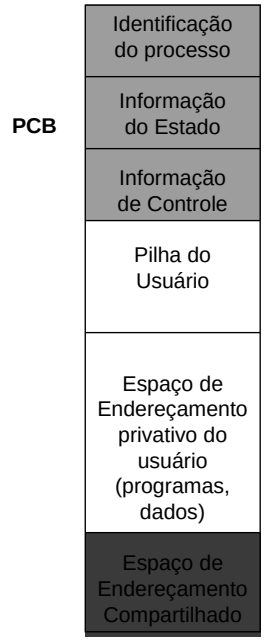


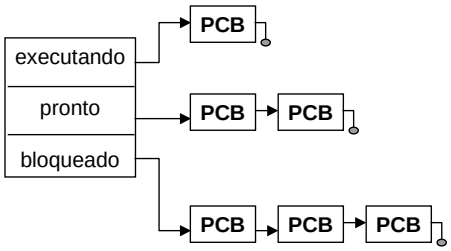
DESCRIÇÃO DE PROCESSO

- PROCESSOS E RECURSOS



Processo de usuário na memória



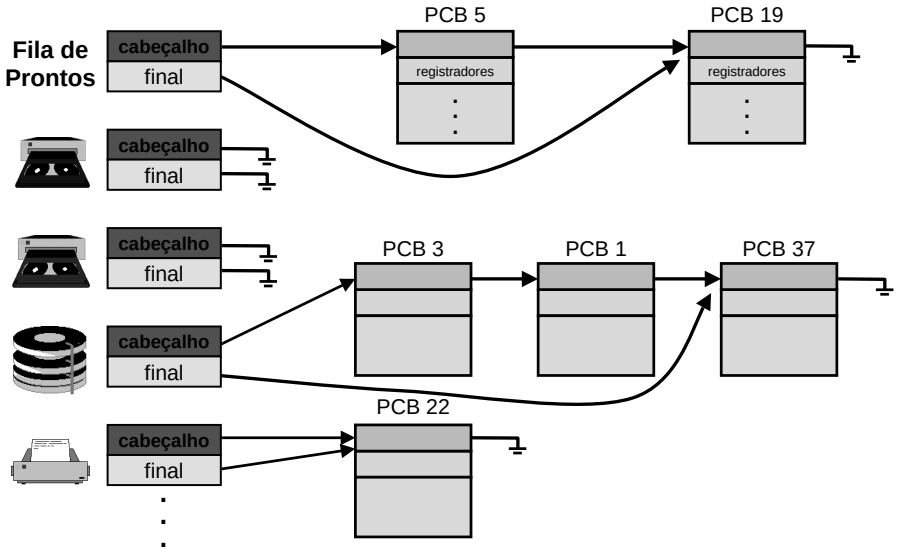


Estrutura com lista de processos

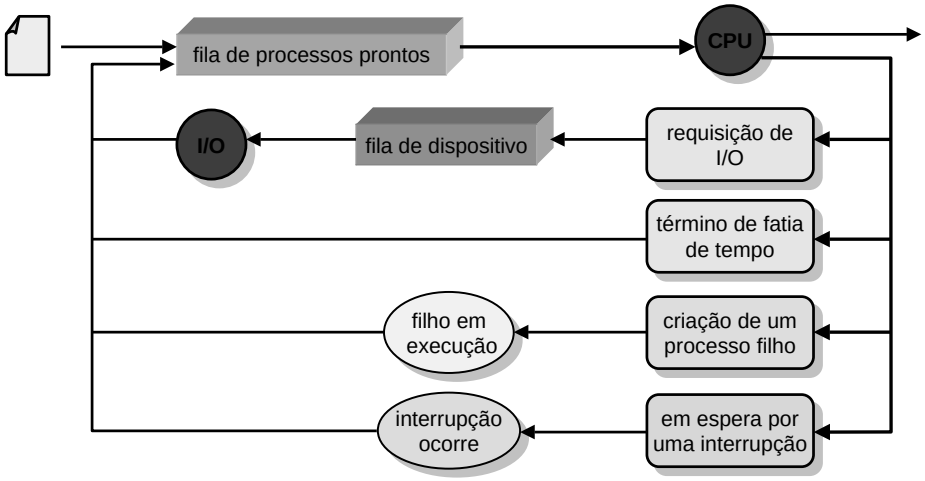
Filas de Escalonamento de Processos

- Fila de serviços – Conjunto de todos os processos no sistema.
- Fila de Processos Prontos – Conjunto de todos os processos residentes na memória, prontos, a espera da execução.
- Filas de Dispositivos – Conjunto dos processos que estão a espera de alguma operação de I/O.
- Os processos migram entre as várias filas.

Fila de Prontos e diversas Filas de Dispositivos de I/O



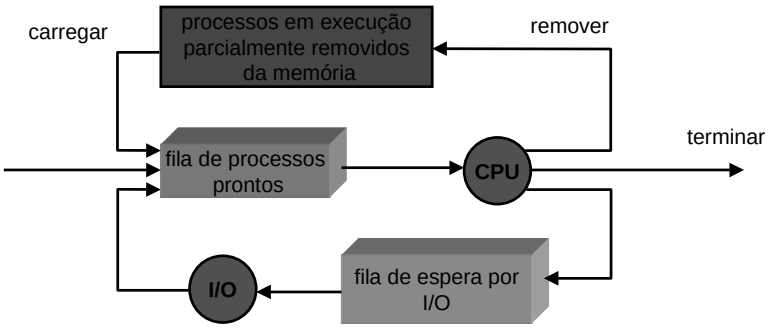
Representação do escalonamento de processos



Escalonadores

- Escalonador de Processos (Long-term scheduler) – seleciona quais processos devem ser trazidos para a fila de prontos.
- Escalonador da CPU (Short-term scheduler) – seleciona qual o próximo processo a tomar posse da CPU.

Adição do Escalonador Intermediário (Medium Term Scheduler)



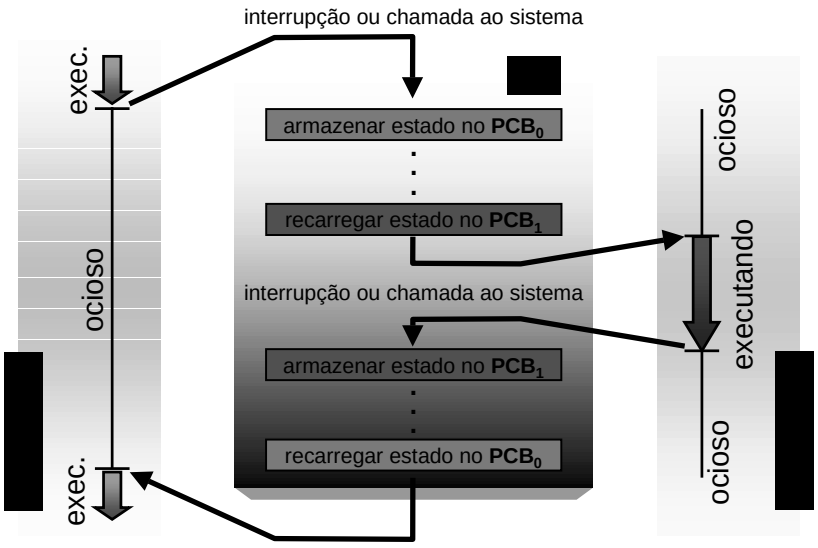
Escalonadores (Cont.)

- Escalonador da CPU é invocado muito freqüentemente (milissegundos).
  $\Rightarrow$ (precisa ser rápido).
- Escalonador de processos é invocado com muito pouca freqüência (segundos, minutos).
  $\Rightarrow$ (pode ser lento).
- O escalonador de processos controla o grau de multiprogramação do sistema.
- Processos podem ser classificados como:
  - I/O-*bound process* – gastam mais tempo fazendo I/O do que computando dados, possui poucas rajadas de CPU.
  - CPU-*bound process* – gastam mais tempo computando dados; realiza poucas, porém longas rajadas de CPU.

Mudança de Contexto

- Quando a CPU é chaveada para outro processo, o sistema deve salvar o estado do processo antigo e carregar o estado do novo processo.
- Mudança de contexto implica overhead; o sistema não realiza nenhum trabalho util durante os chaveamentos.
- O tempo consumido é dependente do suporte de hardware fornecido.

Chaveamento da CPU entre dois Processos



criação de processos

- **atribuir um pid único**
- **alocar as estruturas de dados associadas a um processo**
- **alocar o espaço necessário em memória**
- **PCB deve ser iniciado**
- **consistência das listas do SO devem ser mantidas**
- **outras estruturas devem ser iniciadas (ex.: account)**

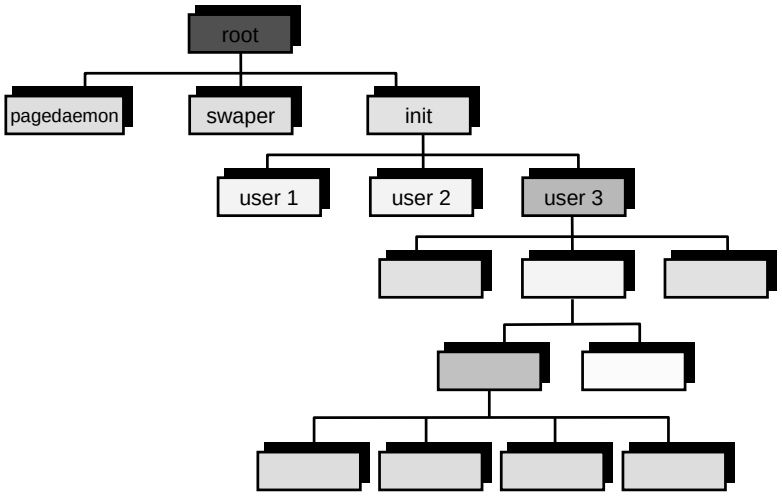
Criação de Processos

- **Processos pais criam processos filhos, os quais, por sua vez, criam novos processos, formando uma árvore de processos.**
- **Compartilhamento de recursos:**
  - **Pais e filhos compartilham os mesmos recursos.**
  - **Filhos compartilham um subconjunto dos recusos do pai.**
  - **Pais e filhos não compartilham algum recurso.**
- **Execução:**
  - **Pai e filhos executam concorrentemente.**
  - **Pai aguarda até os filhos terminarem.**

Criação de processos (Cont.)

- **Espaço de endereçamento**
  - **Filho é uma duplicata do pai.**
  - **Filho tem um novo programa carregado.**
- **Exemplos doUNIX**
  - **A chamada ao sistema** fork **cria novos processos.**
  - **A chamada ao sistema** execve **usada após uma** fork **para sobrepor o espaço de endereçamento do processo com um novo programa.**

Uma árvore de processos em um sistema Unix típico



Finalização de Processos

- **Processos executam a última linha e chamam o sistema operacional para controlar a terminação. (**exit**)**
  - **Saída de dados do filho para o pai (via** wait **).**
  - **Recursos do processo são desalocados pelo sistema operacional.**
- **Pais podem terminar a execução de processos filhos. (**abort**).**
  - **Filhos excederam os recursos alocados.**
  - **Tarefa de encumbência do filho não é mais necessária.**
  - **Pai está terminando.**
    - **Sistema operacional não permite que o filho continue se o seu pai terminou.**
    - **Terminação em cascata.**

Processos cooperativos

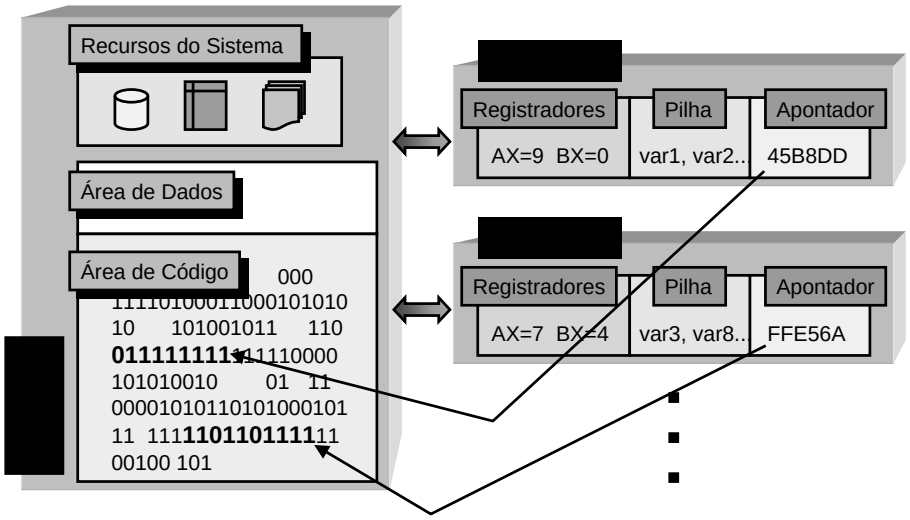
- Processos *independentes* não podem afetar a execução uns dos outros.
- Processos *cooperativos* podem afetar ou ser afetados pela execução de um outro processo envolvido.
- Vantagens da cooperação entre processos:
  - Compartilhamento de informações
  - Aumento da velocidade de computação (speed-up)
  - Modularidade
  - Conveniência

Threads (Fluxos de execução)

- Um *thread* (ou *processo leve*) é uma unidade basica de utilização da CPU. Ela consiste em:
  - apontador de instruções
  - conjunto dos registradores
  - espaço de pilha
- Uma thread compartilha com threads irmãs:
  - área de código
  - área de dados
  - recursos do sistema operacional

  (coletivamente conhecidos como tarefa)
- Um processo tradicional é equivalente a uma tarefa com uma única thread.

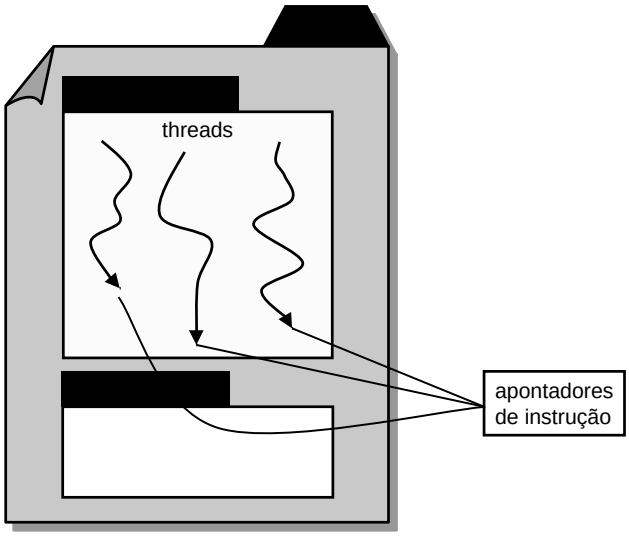
Threads interagindo com Tarefa



Threads (Cont.)

- **Em uma tarefa dotada de múltiplos fluxos de execução, enquanto um fluxo está bloqueado esperando, um outro fluxo na mesma tarefa pode continuar rodando.**
  - **Cooperação de multiplas threads em uma mesma tarefa aumenta o throughput e performance.**
  - **Aplicações que requerem o compartilhamento de buffers (por exemplo, pordutores e consumidores) se beneficiam da utilização de threads.**
- **O mecanismo de threads permite que processos seqüenciais sejam executados paralelamente, apesar de poderem fazer chamadas ao sistema que bloqueiam processos.**
- **Threads oferecidas pelo kernel (Mach e OS/2).**
- **Threads em nível de usuários; suportadas acima do kernel, via chamadas a um conjunto de bibliotecas ao nível de usuário (Posix Pthreads).**
- **Enfoque híbrido implementa tanto threads em nível de usuário quanto threads suportadas pelo kernel (Solaris 2).**

Múltiplas threads em uma tarefa:



UNIX

DIAGRAMA DE TRANSIÇÃO DE ESTADOS