Sistemas Operacionais

Processos

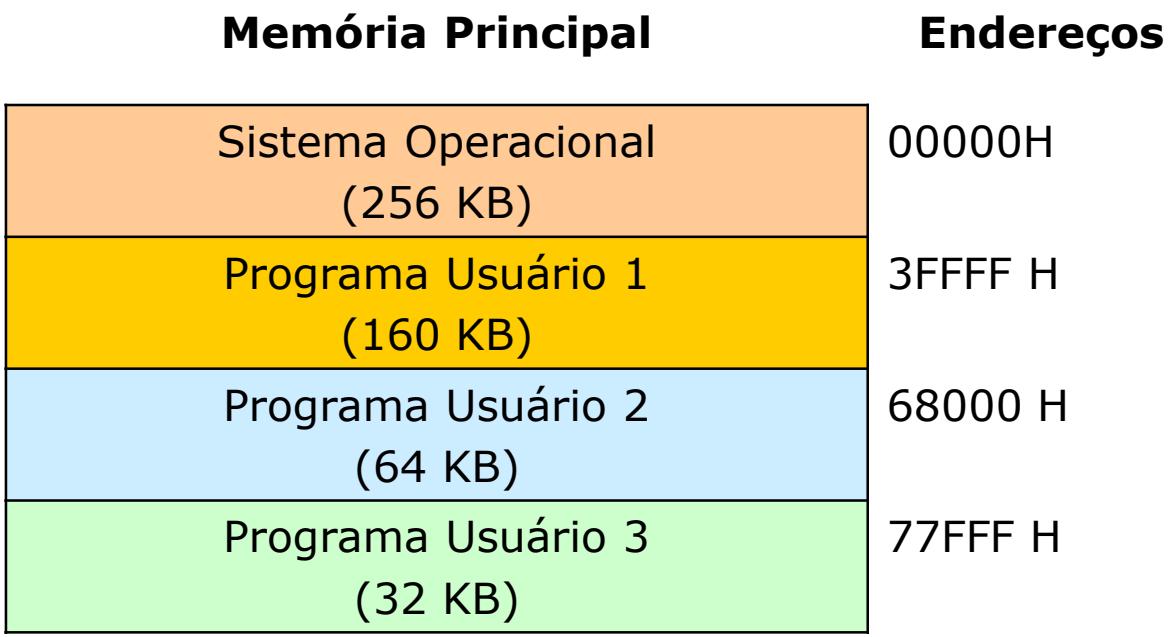
Sumário

- Introdução
- Multiprogramação
- Processos
- Escalonamento de processos
- Operações nos processos
- Comunicação entre processos
- Mais alguns conceitos...
- Referências Bibliográficas

Introdução

- Primeiros sistemas: um programa por vez
- Atualmente **Multiprogramação**
  - muitos programas carregados na memória executados de forma concorrentemente
- Processos que executam concorrentemente, selecionados pela CPU a fim de tornar o computador mais produtivo

Introdução

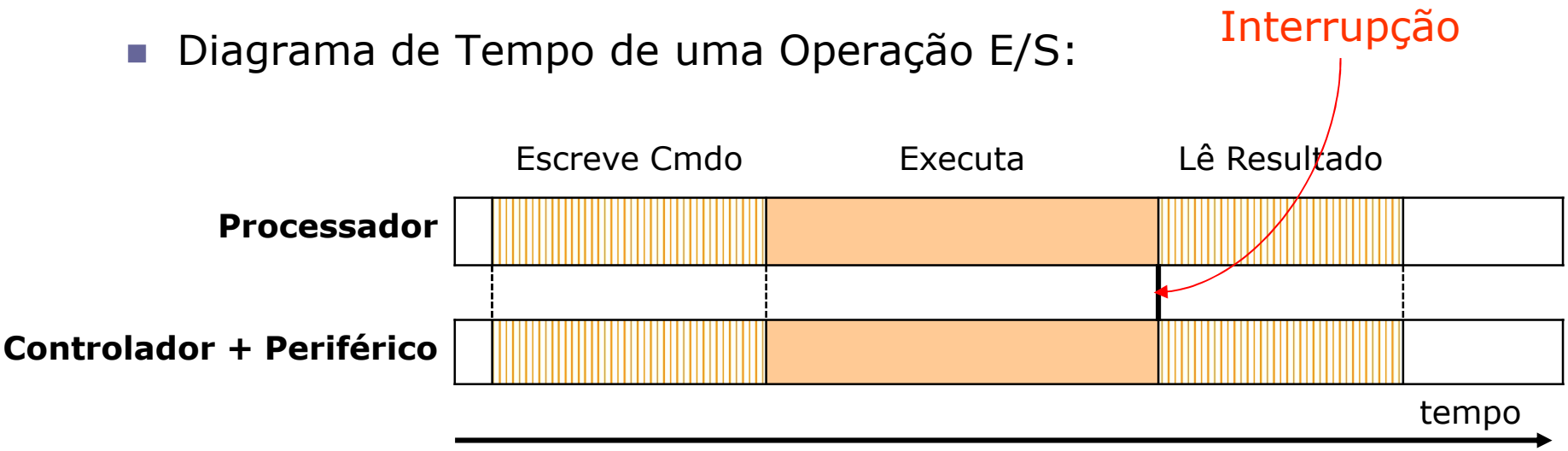

Memória Principal
Endereços
Sistema Operacional
(256 KB)
00000H
Programa Usuário 1
(160 KB)
3FFFF H
Programa Usuário 2
(64 KB)
68000 H
Programa Usuário 3
(32 KB)
77FFF H

Multiprogramação

- Mecanismo de Interrupções
  - Permite que um controlador de periférico chame a atenção do processador
  - Interrupção sinaliza a ocorrência de algum evento
    - Desvia a execução da posição atual para uma outra rotina
    - tratador de interrupção: realiza as ações necessárias em função da ocorrência da interrupção; quando termina volta à rotina interrompida ("sem que essa perceba")
    - Pode ser ativada por hardware ou software
  - Alguns processadores salvam automaticamente os registradores quando ocorre uma interrupção; outros, salvam apenas alguns e a rotina de interrupção encarrega-se de salvar os demais.

Multiprogramação

- Mecanismo de Interrupções
  - Controlador de periférico: conecta o periférico ao processador; traduz sinais ao dispositivo.
  - Processador: ler, escrever dados, ler status do dispositivo, reiniciar, escrever comandos.
  - Diagrama de Tempo de uma Operação E/S:



Multiprogramação

- Mecanismo de Interrupções
  - As interrupções possuem um tipo (0..255) definido pelos projetistas do sistema
    - Há uma relação de prioridades entre as interrupções
  - Existem momentos em que não podem ocorrer interrupções (habilitar/desabilitar interrupções)
    - Evitar acesso a valores incorretos

Multiprogramação

- Processo
  - Unidade de Trabalho
  - Programa em execução
  - Processos
    - Sistema Operacional
    - Usuário
  - CPU Multiplexada entre os processos
  - Um programa por si só não é um processo
    - Programa
      - Entidade Passiva (disco)
    - Processo
      - Entidade Ativa
        - *Program Counter* indica próxima instrução
        - Possui um conjunto de recursos associados

Multiprogramação

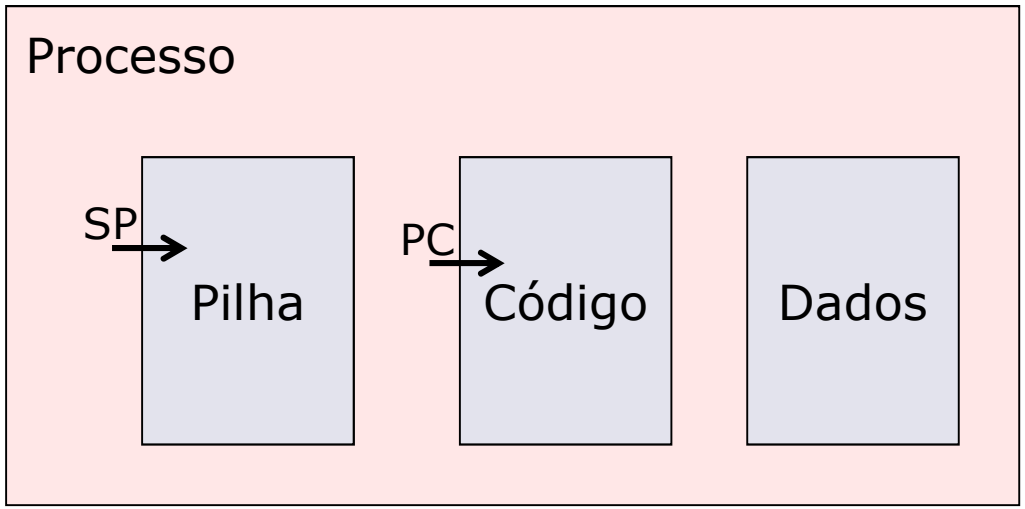
- Proteção entre processos
  - Dois modos de operação
    - Usuário
      - Algumas instruções não podem ser executadas
    - Supervisor
      - Instruções privilegiadas
  - As interrupções chaveiam o processador no modo supervisor
  - Proteção de periféricos
  - Proteção de memória

Processos

- Processo é um conceito utilizado em sistemas operacionais para identificar um elemento que concorre à execução
  - *"Um mesmo programa pode estar sendo executado por vários usuários, ao mesmo tempo, mas para cada usuário existe um processo"*
  - Cada processo trabalha sobre uma área de memória privativa
- Um Programa é uma seqüência de instruções
  - Entidade Passiva
- Um Processo altera seu estado à medida que executa um programa
  - Entidade Ativa

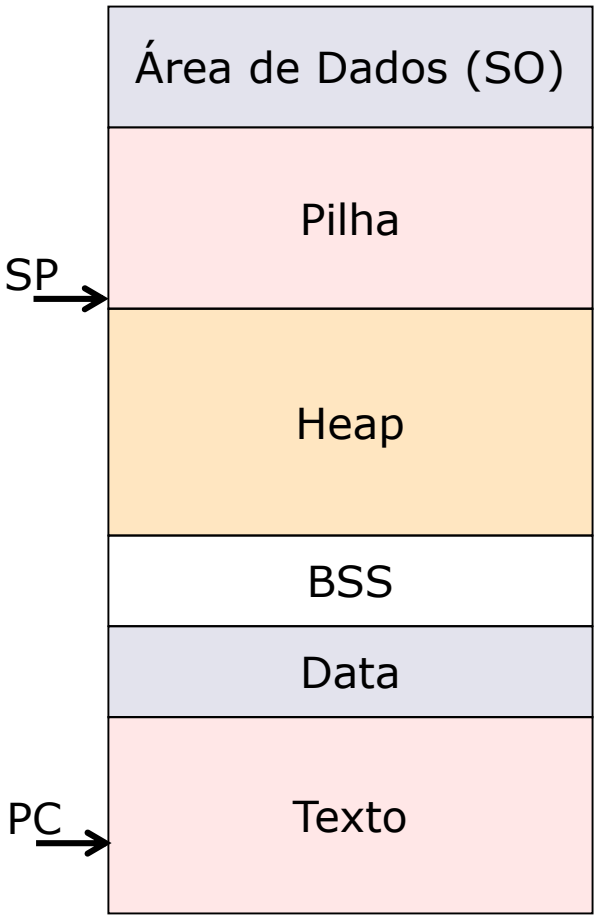
Processos

- É representado pelo:
  - Espaço de Endereçamento (armazenamento da imagem do processo)
  - Estruturas Internas do Sistema (áreas de memórias, tabelas internas...)
  - Contexto de execução (pilhas, dados...)



Modelo de Processo Unix (Linux)

(OLIVEIRA, 2000)



Registros correspondentes à invocação de funcões

Dados alocados dinamicamente usando malloc

Dados não-inicializados

Dados inicializados - Dados alocados estaticamente pelo compilador

Instruções do programa

Estados do Processo

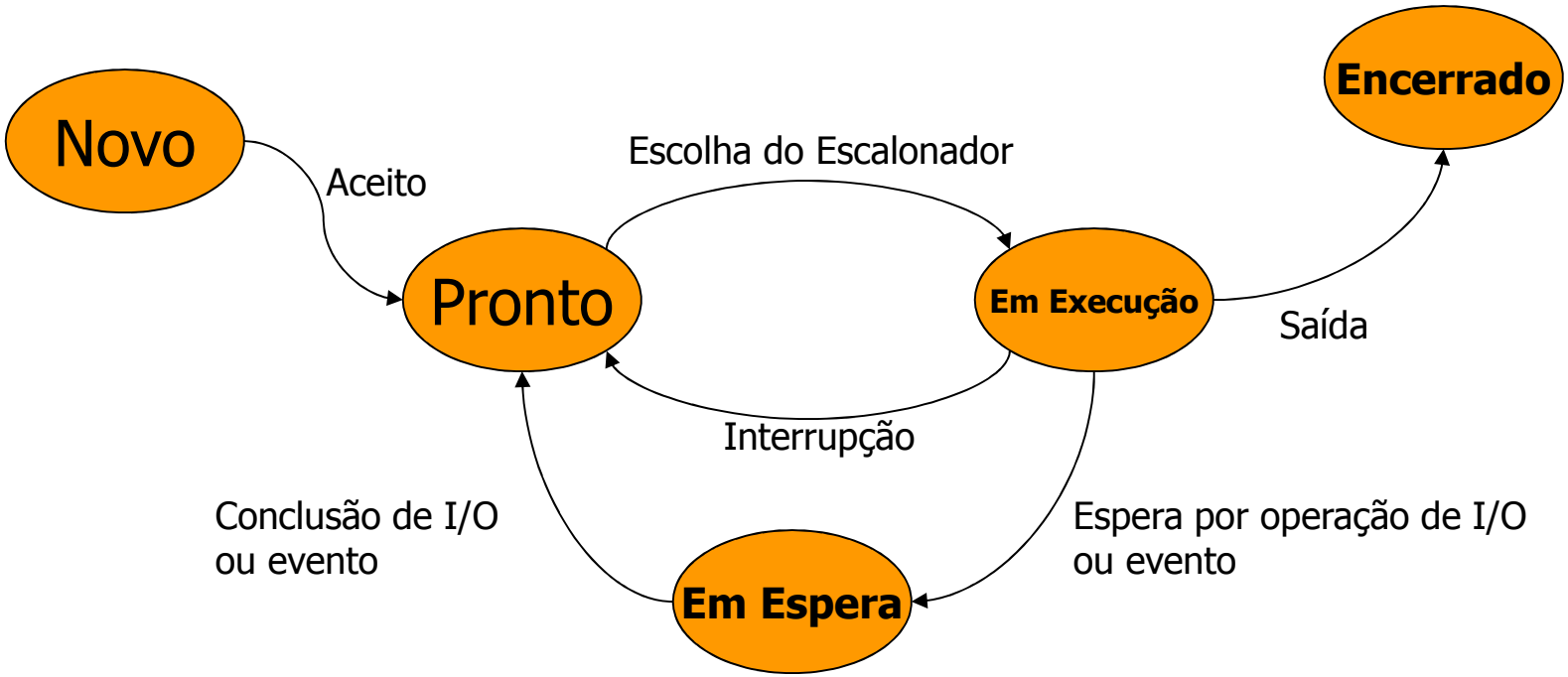
- Novo (*New*)
- Em execução (*Running*)
- Em espera (*Waiting*)
- Pronto (*Ready*)
- Encerrado (*Terminated*)

- Conforme um processo é executado, ele muda de estado

- Apenas um processo executa em qualquer processador a cada instante

- Muitos processos podem estar prontos ou em espera

Diagrama de Estados de um Processo



Processos

- Processos são criados e destruídos
  - Depende do SO
  - Alguns casos
    - Um processo para cada terminal
      - criados na inicialização e destruídos quando o sistema é desligado
    - Um processo para cada sessão de trabalho
    - Criar livremente através de chamadas ao sistema
- Processos do sistema (*daemon*)
- Processos
  - *cpu-bound*
    - Tempo de execução predominantemente definido pelo tempo dos ciclos de processador
  - *i/o-bound*
    - Tempo de execução predominantemente definido pelas operações de E/S

Bloco de Controle de Processo

- **Process Control Block – PCB**
- Cada processo é representado pelo SO por um PCB
- PCB
  - Repositório de informações de cada processo

- Estado do Processo
- Contador de Programa
- Registradores do processador
- Pilha
- Seção de dados (var. globais)
- Seção de texto (código)
- Informações para o escalonamento
- Informações sobre o gerenciamento de memória
- Informações sobre a contabilização
- Informação sobre *status* de E/S

Bloco de Controle de Processo

- ou Bloco Descritor de Processo
  - Exemplo da estrutura do PCB

```
struct desc_proc{
    char     estado_atual;
    int      prioridade;
    unsigned inicio_memoria;
    unsigned tamanho_memoria;
    struct   arquivo arq_aberto[20];
    unsigned tempo_gasto_cpu;
    unsigned proc_pc;
    unsigned proc_sp;
    unsigned proc_acc;
    unsigned proc_rx;
    //Lista de descritores
    struct   desc_proc *prox;
    struct   desc_proc *livre;
    struct   desc_proc *espera_CPU;
    struct   desc_proc *usando_cpu;
}
```

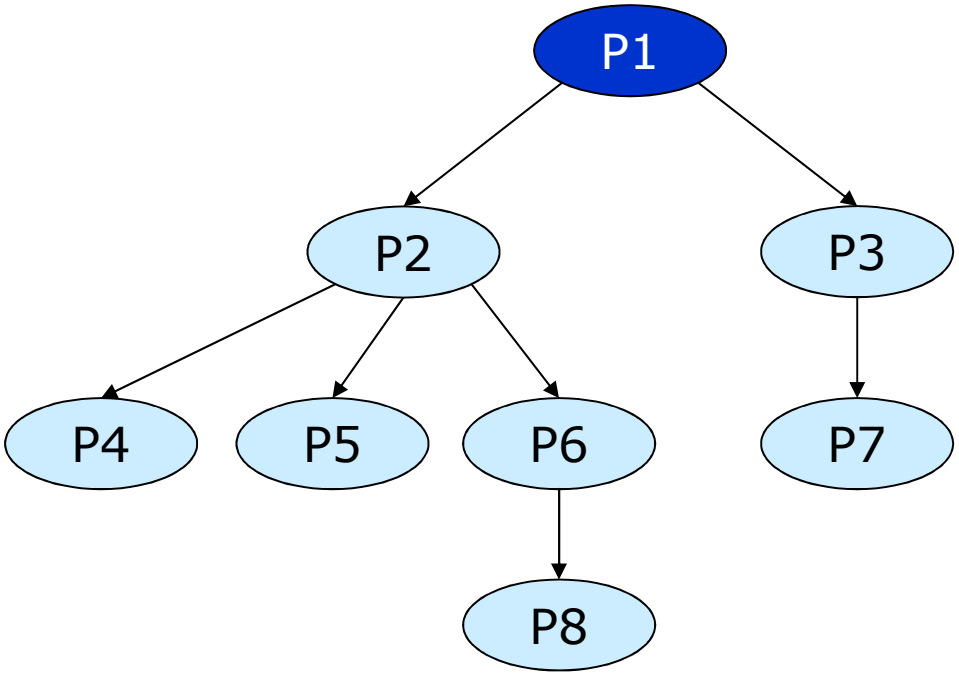
Durante o Ciclo de Vida do Processo, no PCB...

- Criação
  - Alocação de área de memória para código, dados e pilha do processo e estruturas do sistema operacional
  - Inicialização do PCB e inserção na fila
- Execução
  - Execução das instruções da área de código (há interação com o SO)
  - Atualização dos estados e recursos do processo no PCB
- Término
  - Liberação de recursos e estruturas de dados utilizadas

Hierarquia de Processos

- Alguns SO permitem trabalhar com o conceito de grupo de processos
  - Permite aplicar a mesma operação sobre todo o conjunto de processos
    - Direitos do processo
- Processos podem ser criados por outros processos através de chamadas ao sistema
  - Processo que faz a chamada: **Processo Pai**
  - Processo criado: **Processo Filho**

Hierarquia de Processos



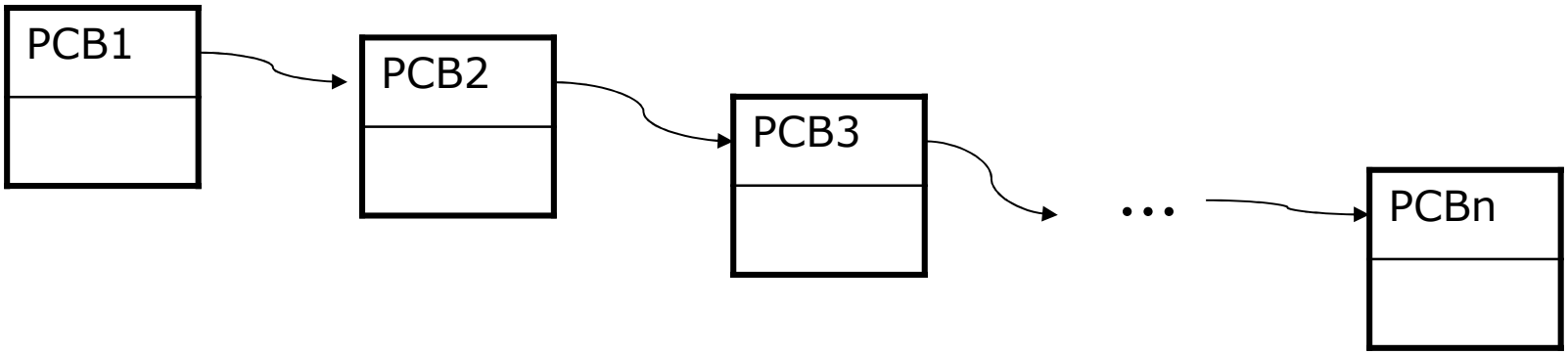
- P1 é o processo inicial do sistema, criado durante a inicialização
- Os demais processos foram criados através de chamadas ao sistema.
- A hierarquia muda com o passar do tempo
- Quando um processo é destruído, o que acontece com seus filhos?

Escalonamento de Processos

- Multiprogramação
  - Processos em execução o tempo todo
  - Otimizar CPU
- Tempo compartilhado
  - Alternar a CPU entre processos (frequentemente)
  - Transparente para usuário
- Uniprocessador
  - Escalonamento
- Filas de Escalonamento
  - Filas de Jobs
    - Todo processo que entra no sistema é colocado nessa fila
  - Filas de Processos Prontos
    - Processos que estão na memória e estão prontos para serem executados
  - Filas de Dispositivos ou I/O (cada dispositivo tem a sua)
    - Lista de processos esperando por um determinado dispositivo
    - Quando o dispositivo está ocupado, o processo é colocado na fila do dispositivo

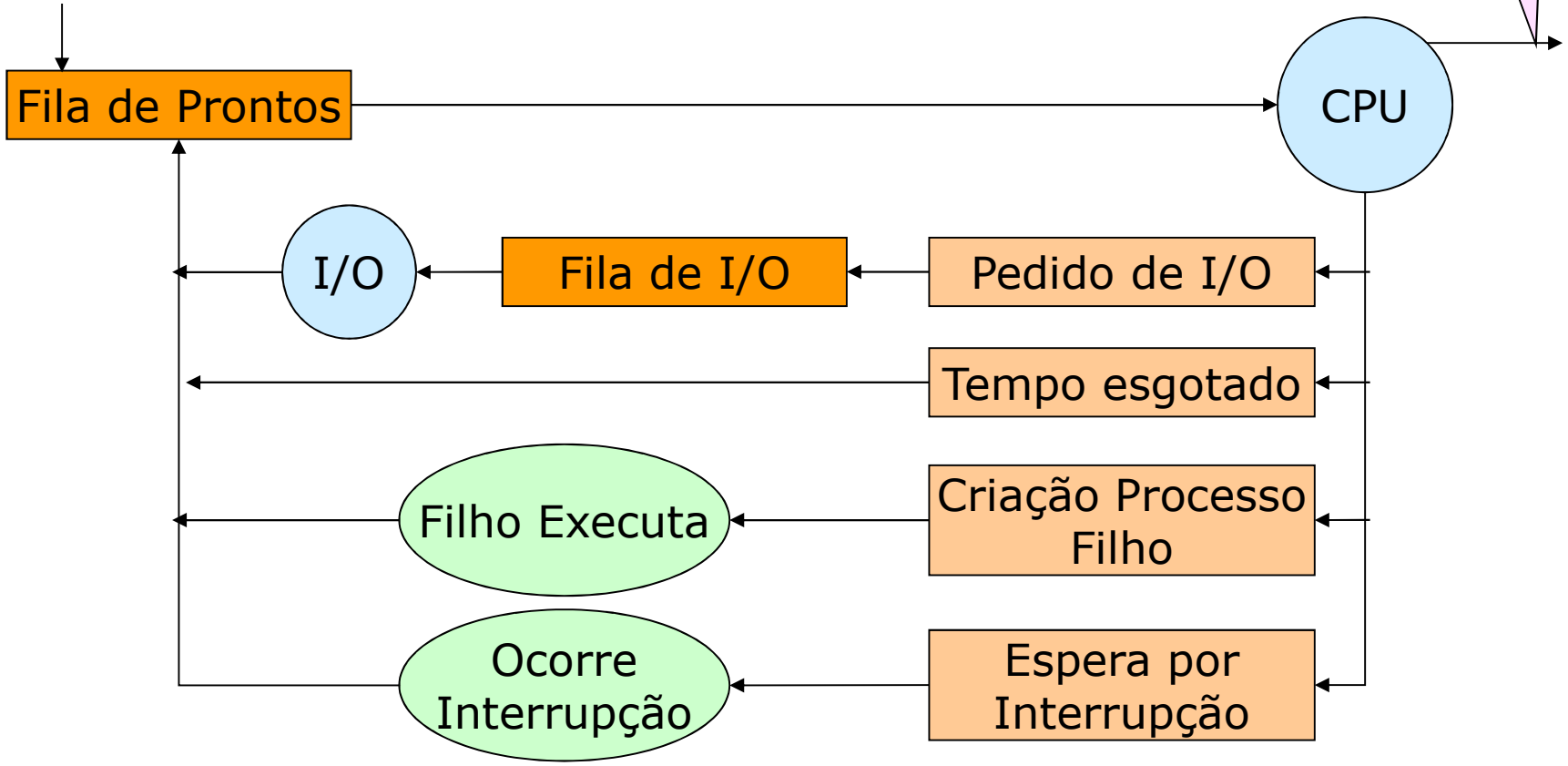
Escalonamento de Processos

- Filas de Escalonamento
  - Lista encadeada
    - Cabeçalho
      - Ponteiros para o início e o fim da lista
    - PCB
      - Ponteiros para o próximo processo da lista



Escalonamento de Processos

- Diagrama de Filas



Escalonamento de Processos

- Escalonador *(scheduler)*
  - Responsável por selecionar o processo (nas várias filas) que será alocado na CPU
  - Processos migram entre as várias filas de escalonamento durante sua vida

Escalonador de Longo Prazo

Escalonador de Curto Prazo

Escalonador de Médio Prazo

Escalonamento de Processos

- Escalonador de Longo Prazo - *Long-Term*
  - O escalonador de longo prazo (*Long-Term*) seleciona processos e os carrega na memória para execução
  - É executado com muito menos freqüência (minutos)
  - Deve selecionar processos com cuidado a fim de balancear a carga do sistema
  - Controla o grau de multiprogramação
    - Número de processos na memória
    - Grau estável
      - Taxa média de criação de processos = Taxa média de saída de processos no sistema
  - Chamado somente quando os processos saem do sistema

Escalonamento de Processos

- Escalonador de Longo Prazo - *Long-Term*
  - Seleciona uma boa combinação de
    - Processos I/O Bound
    - Processos CPU Bound
  - Muitos I/O Bound: Fila de prontos vazia
  - Muitos CPU Bound: Fila de I/O vazia; dispositivos sem uso

Escalonamento de Processos

- Escalonador de Curto Prazo - *Short-Term*
  - Seleciona processos a partir da **fila de prontos**
    - **Processos que concorrem á CPU**
  - A principal diferença entre escalonadores de Curto e Longo Prazo é a freqüência de execução
  - Deve selecionar novos processos com bastante freqüência (milisegundos)
  - Deve ser bastante rápido pois pode-se perder mais de 10% do tempo somente com escalonamento

Escalonamento de Processos

- Escalonador de Médio Prazo - *Medium-Term*
  - Nível intermediário de escalonamento
  - Usa a idéia de que às vezes pode-se ter vantagens em remover o processo da memória, **reduzindo o nível de multiprogramação**
    - Posteriormente retorna ao ponto onde parou a execução
  - Útil para
    - Melhorar a combinação de processos
    - Mudança de requisitos durante a execução dos processos compromete a memória disponível
  - *Swapping*

Operações nos Processos

- Processos podem ser Criados ou Excluídos dinamicamente

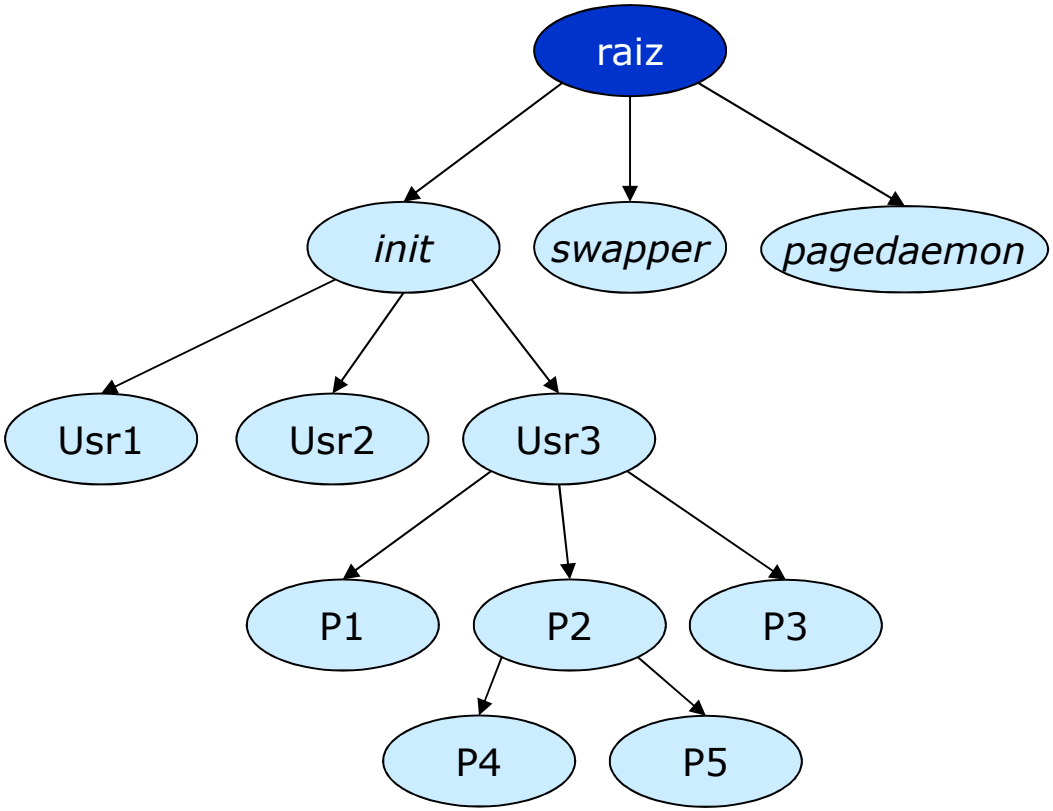
- Criação de Processos
  - Um processo pode criar outros processos (*system call create-process*)
  - Processo criador: Pai (*Parent*)
  - Processo criado: Filho (*Children*)
  - Processo filho pode obter recursos de seu pai ou diretamente do SO
    - Se limitado a um subconjunto do Pai → evita sobre-carga do sistema devido a criação de muitos processos

Operações nos Processos

- Criação de Processos
  - Quando um processo cria outro, em termos de execução, pode ocorrer:
    - O pai executa concorrentemente com o filho
    - O pai espera que os filhos terminem sua execução
  - Quando um processo cria outro, em relação ao espaço de memória, pode ocorrer:
    - O processo filho é uma duplicata do pai
    - O processo filho tem um programa carregado em si
  - No Unix, um novo processo é criado com a chamada de sistema **fork.** Este novo processo é uma cópia do pai. Usando a chamada **execve** depois de um **fork**, é executado um novo programa que é carregado na memória, destruindo a imagem do processo que o chamou.

Operações nos Processos

- Criação de Processos
  - Árvore de processos no Unix



Operações nos Processos

- Término de Processos
  - O processo acaba quando é executada a última linha de comando do programa
  - Este retorna dados a seu processo pai
  - O pai pode terminar a execução do filho:
    - O filho excede a utilização de recursos
    - A tarefa realizada pelo filho não é mais necessária
    - O pai está sendo terminado

Desalocar memória, arquivos abertos, *buffers*

Operações nos Processos

- Comandos
  - fork
    - Permite a criação de um segundo fluxo de execução
    - Fornecer o nome da subrotina ou programa
    - Gerência do processador cria estruturas de dados necessárias e insere o processo na fila de prontos
      - O espaço de endereçamento deve ser igual ao do processo que o criou
  - exit
    - Quem o executa é imediatamente terminado
    - Executado por uma chamada de sistema
  - wait
    - Um fluxo de execução espera outro terminar
    - Ocorre o bloqueio do processo o qual é inserido em uma fila de processos bloqueados a espera do determinado processo

Operações nos Processos

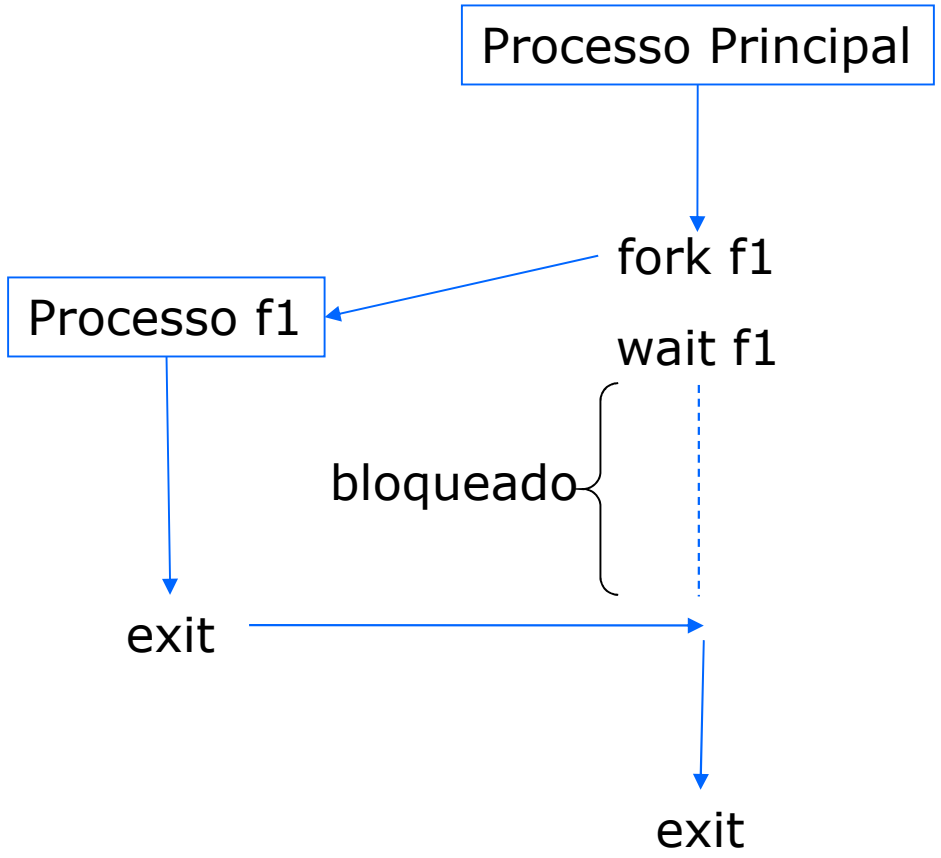
```
int filho()
{
    printf("Eu sou o filho\n");
    exit(0);
}

int main()
{
    pid_t procID;

    procID = fork();

    if (procID == 0)
         {
          printf("Processo filho - para o filho o fork devolveu %d\n", procID);
          printf("Processo filho - PID = %d\n", getpid());
          filho();
          return 1;
         }
         else
         {
          wait(NULL);
          printf("Processo Pai - para mim o fork devolveu %d\n", procID);
          printf("Processo Pai - PID = %d\n", getpid());
          return 1;
         }
}
```

Operações nos Processos



Comunicação entre processos

- Trocas de Mensagens
  - Uso das primitivas de envio (*send*) e recebimento (*receive*)
- Sinais
  - São interrupções de software que notificam ao processador um evento que ocorreu
  - Não permitem trocar dados
  - Um processo, ao receber um sinal, pode:
    - Capturar
    - Ignorar
    - Mascarar (bloquear)

Mais alguns conceitos...

Programação Concorrente

Chaveamento de Contexto

Programação Concorrente

"Um programa concorrente é executado simultaneamente por diversos processos que cooperam entre si" (OLIVEIRA, 2000)

- É necessária a interação entre os processos para que se caracterize a concorrência
  - Arquivos
  - Variáveis compartilhadas
  - Trocas de mensagens
- Processos concorrem a recursos

Chaveamento de Contexto

- Troca de Contexto
  - Alternar a CPU para outro processo

Salvar estado antigo e **carregar** novo estado

- *Kernel*
  - Quando um processo perde a CPU, seu contexto deve ser salvo
  - Para executar outro processo, seu contexto deve ser carregado na CPU
- Contexto
  - Está no PCB (registradores do processador, estado, gerenciamento de mémória)
- O chaveamento de contexto resulta em puro *overhead*
- Valores típicos estão entre 1 e 1000 microsegundos (varia conforme o HW utilizado)

Referências Bibliográficas

- SILBERSCHATZ, Abraham; GALVIN, Peter; GAGNE, Greg. Sistemas Operacionais: com Java. Rio de Janeiro: Elsevier, 2004.

- OLIVEIRA, R. S.; CARISSIMI, A. S.; TOSCANI, S. T. Sistemas Operacionais. Porto Alegre: Sagra-Luzzatto, 2000.