DCC062 – Sistemas Operacionais

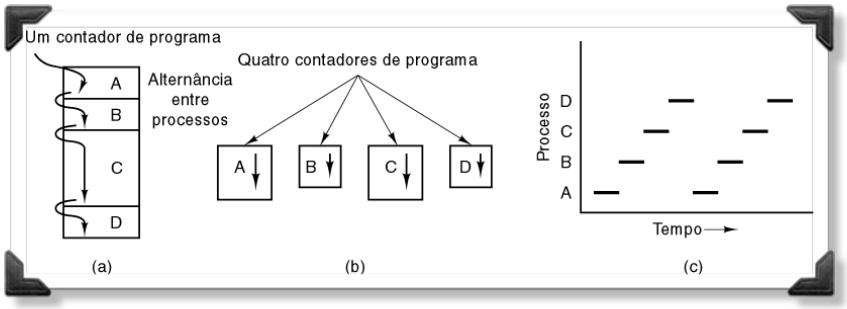
Cap. 2 – Processos
Parte 1

Prof. Marcelo Moreno
moreno@ice.ufjf.br

Processos - Motivação

- Necessidade de gerenciamento dos programas instanciados para execução
- Multiprogramação
  - Pseudo-paralelismo
- Multiprocessamento
- Modelo de processo
  - Processos sequenciais
  - Programa em execução acompanhado de seus valores para contador de programa, registradores e variáveis
  - CPU é trocada de um processo para outro

Multiprogramação



Processos

- Alternância de processos
  - Comandada pelo escalonador de processos do sistema operacional
  - Não somente processos de usuário desejam utilizar CPU
  - Eventos assíncronos do sistema necessitam tratamento de processamento
- Relação Programa x Processo
  - Receita de bolo x Fazer o bolo
  - Multiprogramação?

Criação de processos

- **Principais eventos que levam à criação de processos**
  - **Início do sistema**
  - **Execução de chamada ao sistema de criação de processos**
  - **Solicitação do usuário para criar um novo processo**
  - **Início de um job em lote**
- **Independente do caso, chamada de sistema para criação do processo**
  - **fork(), CreateProcess()**
  - **Espaços de endereçamento distintos**

Término de Processos

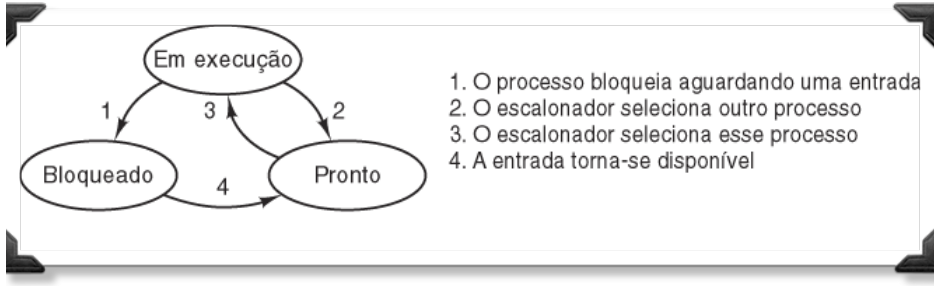
- **Condições que levam ao término de processos**
  - **Saída normal (voluntária)**
  - **Saída por erro (voluntária)**
  - **Erro fatal (involuntário)**
  - **Cancelamento por um outro processo (involuntário)**

Hierarquia de Processos

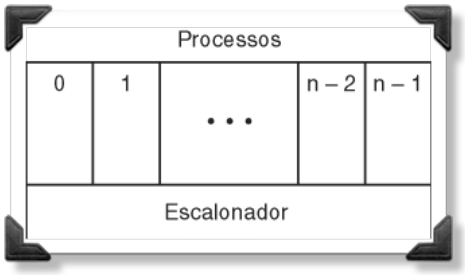
- **Pai cria um processo filho, processo filho pode criar outros processos**
- **Alguns sistemas formam uma hierarquia**
  - **UNIX chama isso de "grupo de processos"**
  - **Windows não possui o conceito de hierarquia de processos**
    - Todos os processos são criados em mesmo nível

Estados de Processos

cat chapter1 chapter2 chapter3 | grep tree



Estados de Processos



Implementação de Processos

- **Tabela de processos**
- **Estrutura processo**

Gerenciamento de processos	Gerenciamento de memória	Gerenciamento de arquivos
Registradores	Ponteiro para o segmento de código	Diretório-raiz
Contador de programa	Ponteiro para o segmento de dados	Diretório de trabalho
Palavra de estado do programa	Ponteiro para o segmento de pilha	Descritores de arquivos
Ponteiro de pilha		Identificador (ID) do usuário
Estado do processo		Identificador (ID) do grupo
Prioridade		
Parâmetros de escalonamento		
Identificador (ID) do processo		
Processo pai		
Grupo do processo		
Sinais		
Momento em que o processo iniciou		
Tempo usado da CPU		
Tempo de CPU do filho		
Momento do próximo alarme		

Implementação de Processos

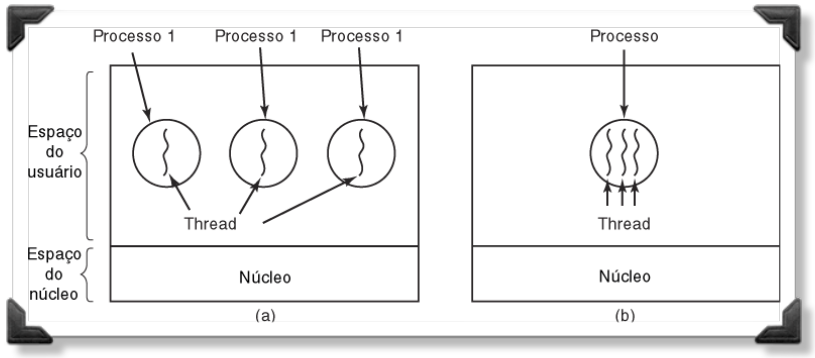
- **Vetor de Interrupções**

1. O hardware empilha o contador de programa etc.
2. O hardware carrega o novo contador de programa a partir do vetor de interrupção.
3. O procedimento em linguagem de montagem salva os registradores.
4. O procedimento em linguagem de montagem configura uma nova pilha.
5. O serviço de interrupção em C executa (em geral lê e armazena temporariamente a entrada).
6. O escalonador decide qual processo é o próximo a executar.
7. O procedimento em C retorna para o código em linguagem de montagem.
8. O procedimento em linguagem de montagem inicia o novo processo atual.

Threads

- **Processo agrupa recursos para uma linha de execução (thread)**
  - **Uma thread possui um contador de programa**
  - **É possível termos várias threads compartilhando recursos de um mesmo processo?**
- **Programação multithread**
- **Modelo de Threads**
  - **Compartilham o mesmo espaço de endereçamento**
  - **Lightweight Processes - processos leves**

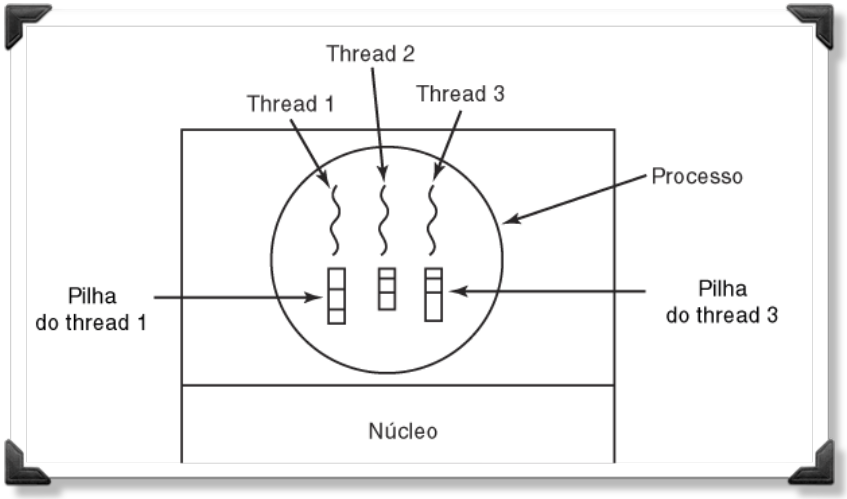
Processos x Threads



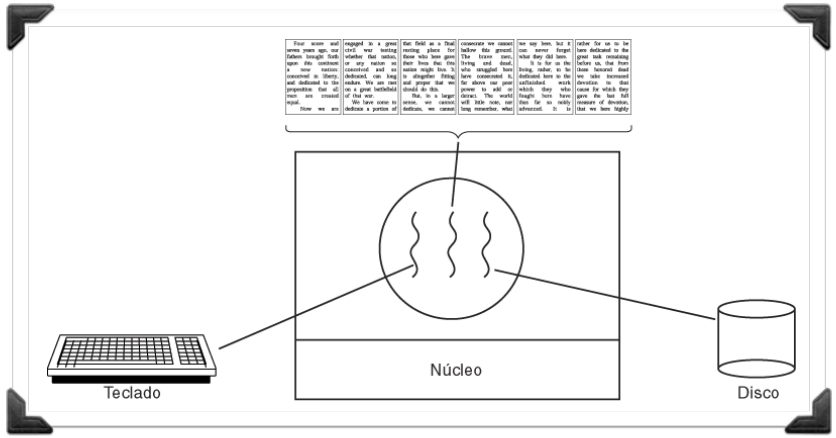
Processos x Threads

Itens por processo	Itens por thread
Espaço de endereçamento	Contador de programa
Variáveis globais	Registradores
Arquivos abertos	Pilha
Processos filhos	Estado
Alarmes pendentes	
Sinais e tratadores de sinais	
Informação de contabilidade	

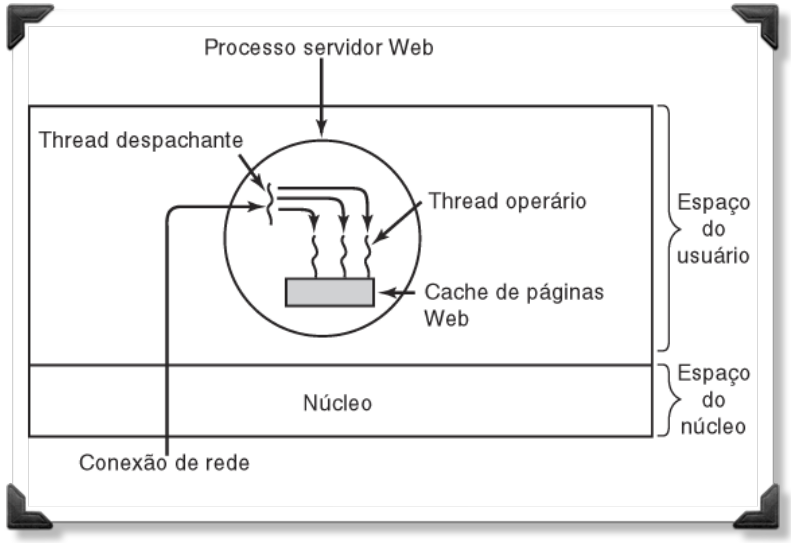
Modelo de Thread



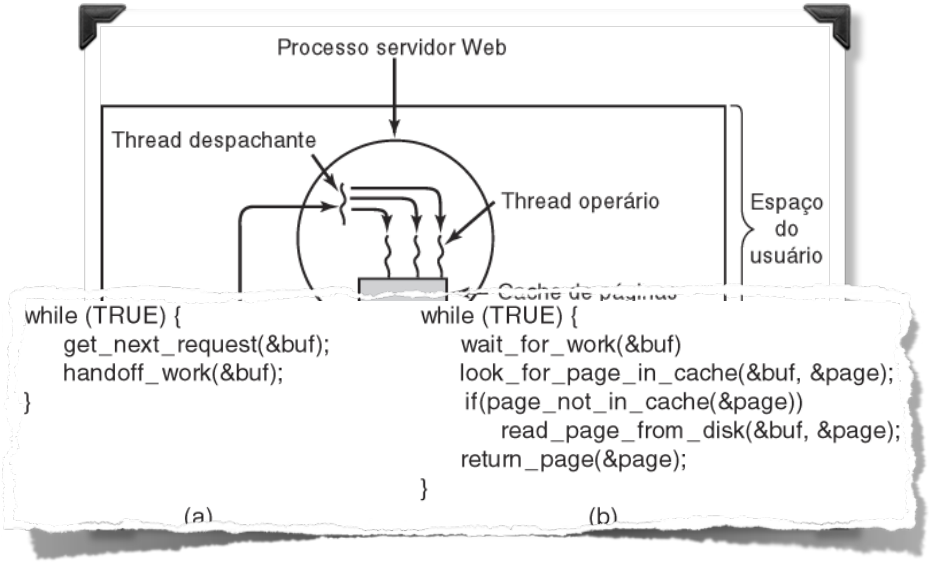
Uso de Threads



Uso de Threads



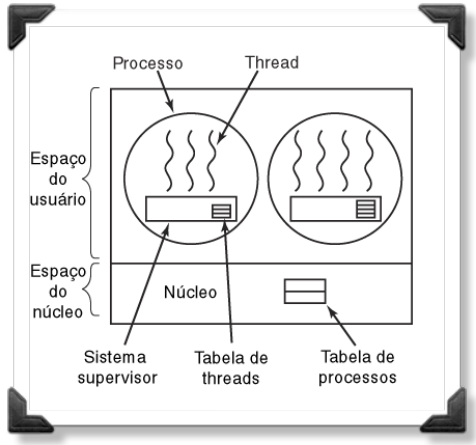
Uso de Threads



Uso de Threads

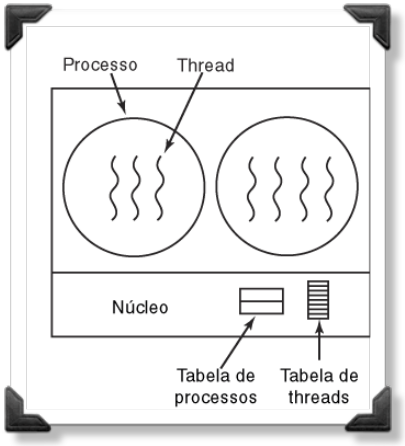
Modelo	Características
Threads	Paralelismo, chamadas ao sistema com bloqueio
Processo monothread	Sem paralelismo, chamadas ao sistema com bloqueio
Máquina de estados finitos	Paralelismo, chamadas ao sistema sem bloqueio, interrupções

Implementação - Threads de Usuário



- **Vantagens**
  - Trocas de contexto no espaço do usuário
  - Escalonamento é procedimento local
  - Escalonador personalizado
  - Escalabilidade
- **Desvantagens**
  - Chamadas ao sistema com bloquei
  - Thread deve liberar CPU explicitamente
  - Threads são úteis para aplicações com bloqueio (I/O bound)
  - Pouco desempenho para aplicaçõe CPU bound

Implementação - Threads de Núcleo



- **Vantagens**
  - **Não é necessário um supervisor**
  - **Tabela de processos estendida**
  - **Escalonamento de threads**
  - **Chamadas ao sistema com bloqueio**
- **Desvantagens**
  - **Velocidade para criar/destruir uma thread**
  - **Trocas de contexto**
  - **Escalabilidade**

Implementação - Threads híbridas