Sistemas Operacionais

CCP/SIF
UNISUL – Tubarão

Cassio Brodbeck Caporal
cassio{NOSPAM}ostec.com.br

Agenda

- Revisão;
- Resolução do exercício;
- *Threads*;
- Ambientes mono e multithread;
- *Threads* em modo usuário, kernel e híbrida;

Threads

- Até 1970:
  - Processos com um único *thread* (um único programa fazendo parte do contexto do processo);
- Em 1979:
  - Surgimento de processos leves (*lightweight*);
  - Compartilhamento do espaço de endereçamento.

Threads

- A partir de 1980:
  - Utilização comercial;
  - Sistema operacional Mach;
- Processos com partes de código sendo executadas em paralelo, com pouquíssimo *overhead*.

Ambiente monothread

- O que é um programa?
  - Seqüência de instruções, com desvios, condições, estruturas de repetição, procedimentos, funções, etc;
- O processo suporte **somente UM** programa em seu espaço de endereçamento.

Ambiente monothread

- Aplicações concorrentes?
  - Ih! Só com processos independentes ou sub-processos (alto overhead);
- Para cada processo criado, diversas alocações devem ser feitas (overhead novamente);
- Cada um tem seu espaço de endereçamento

Ambiente monothread

- Continuando... cada um tem seu espaço de endereçamento:
  - Dificuldade de comunicação entre processos;
  - Utilização de elementos externos como *pipes*, sinais, memória compartilhada, *UNIX domain sockets*;
- Enfim, coisa do passado! :-)

Ambiente multithread

- Agora sim :-)
- Os programas não são mais associados a processos, e sim a *threads*;
- Compartilhamento do espaço de endereçamento por múltiplos *threads*;
- *Thread* como fluxo de execução de um programa (independência do principal).

Ambiente multithread

- Minimização de alocação de recursos do sistema;
- Diminuição do *overhead* na criação, troca e eliminação de processos;
- Compartilham contexto de software e espaço de endereçamento:
  - *Thread Control Block (TCB)*.

Ambiente multithread

- O TCB armazena, essencialmente, o contexto de hardware;
- Divisão de unidade de alocação de recursos e escalonamento;
- Grande diferença entre processos e threads:
  - Compartilhamento do mesmo espaço de endereçamento.

Ambiente multithread

- Inexistência de proteção de acesso a memória;
  - Necessidade de mecanismos de sincronização e comunicação de *threads*;
- *Multithreads* são muito mais rápidos para criação, troca de contexto, entre outras atividades de processos.

Modelos de Implementação

- Pacotes de *threads*;
- Podem ser implementadas em *user level, kernel level* e *mix* dos dois (híbrido);
- Ausência de padrões:
  - POSIX *threads*;
  - Disponível em implementações UNIX.

Threads em modo usuário

- Implementado pela própria aplicação;
- Biblioteca intermediária para a consolidação do uso;
- O sistema operacional não tem conhecimento de *threads*;
- Vantagem: implementar aplicações mesmo em SO que não suporta *threads*.

Threads em modo usuário

- Desvantagens:
  - O sistema operacional gerencia cada processo como se houvesse um único *thread;*
  - Tratamento individual de sinais;
  - Escalonamento: SO seleciona processos, não *threads*.

Threads em modo *kernel*

- Implementados diretamente no sistema operacional (núcleo);
- Grande problema: baixo desempenho;
  - Por que?
    - Mudança de modos de acesso (*user level* para *kernel level* para *user level*);