UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Aula 3.1: Threads

Conceitos e Implementação

Prof. Rodrigo Campiolo
Prof. Rogério A. Gonçalves[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento de Computação (DACOM)
Campo Mourão, Paraná, Brasil

Ciência de Computação

BCC34G - Sistemas Operacionais

Introdução

Threads

- O que é uma **thread**?
  - um fluxo em execução.
  - um processo leve (*lightweight process*).
  - unidade básica que o SO aloca tempo de processador[a].
- Por que usar threads?
  - Dividir tarefas de uma aplicação (ex: navegador Web).
  - Compartilhamento de espaço de endereçamento.
  - Tempo de criação e finalização.
  - Concorrência e paralelismo.

[a] `https://docs.microsoft.com/en-us/windows/win32/procthread/processes-and-threads`

Threads: Exemplo de uso - Processador de texto

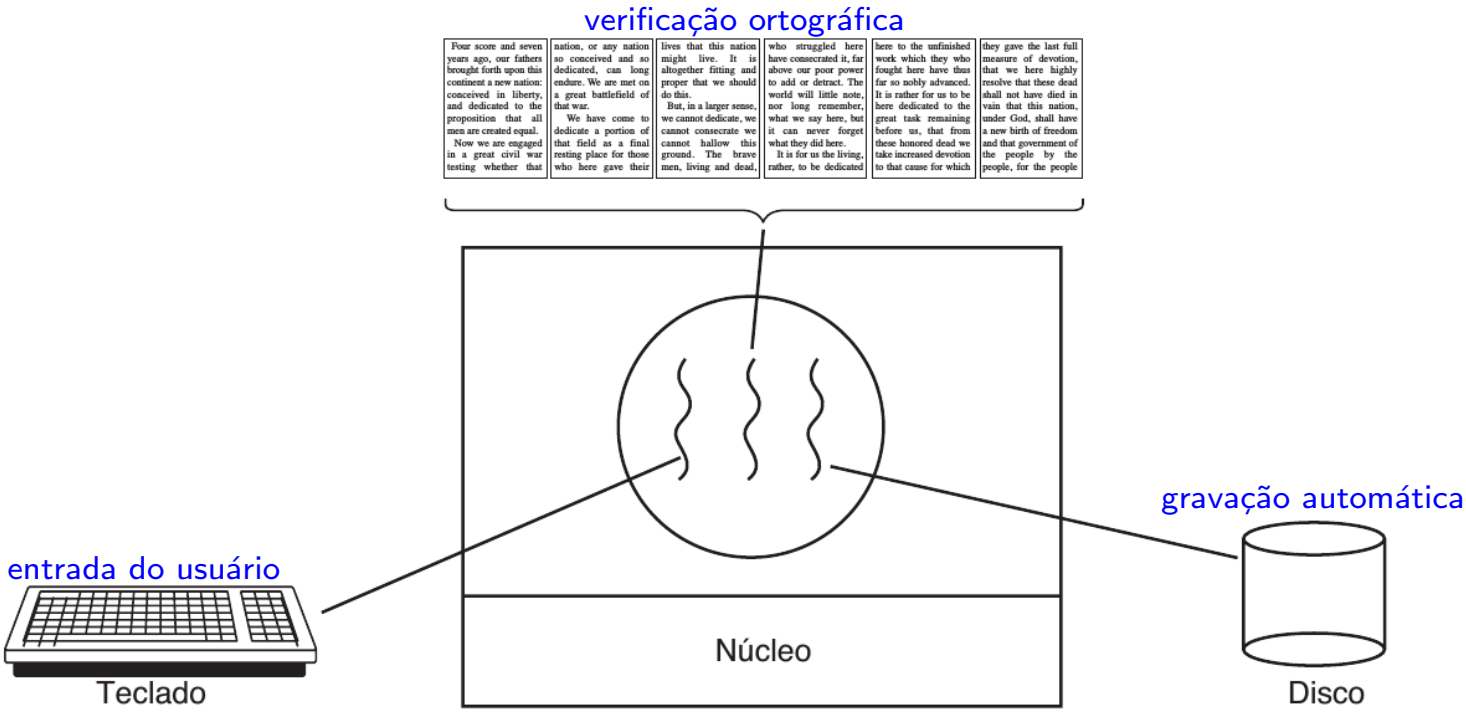


Figura 1: Processador de texto com três threads [3].

Threads: Exemplo de uso - Servidor Web

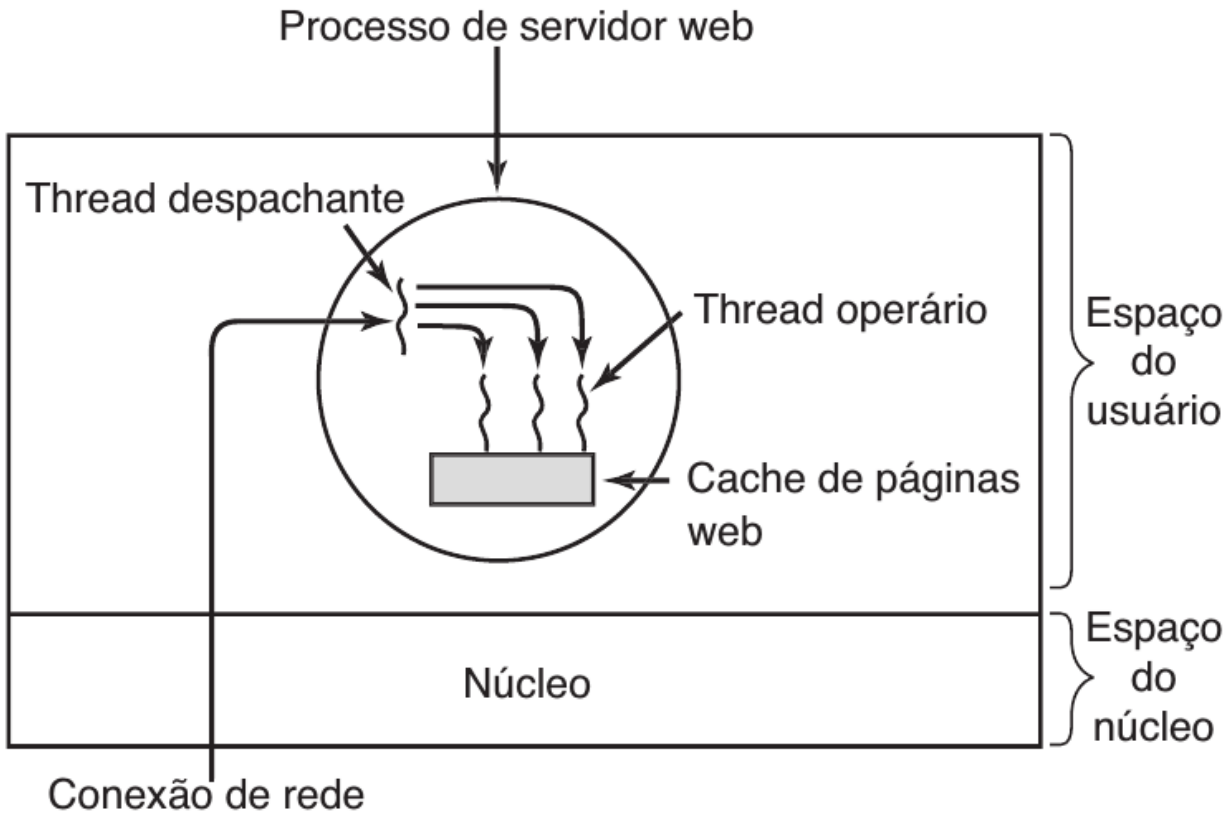


Figura 2: Servidor Web atendendo múltiplas requisições simultâneas [3].

Threads: Implementação

- Modelos clássicos:

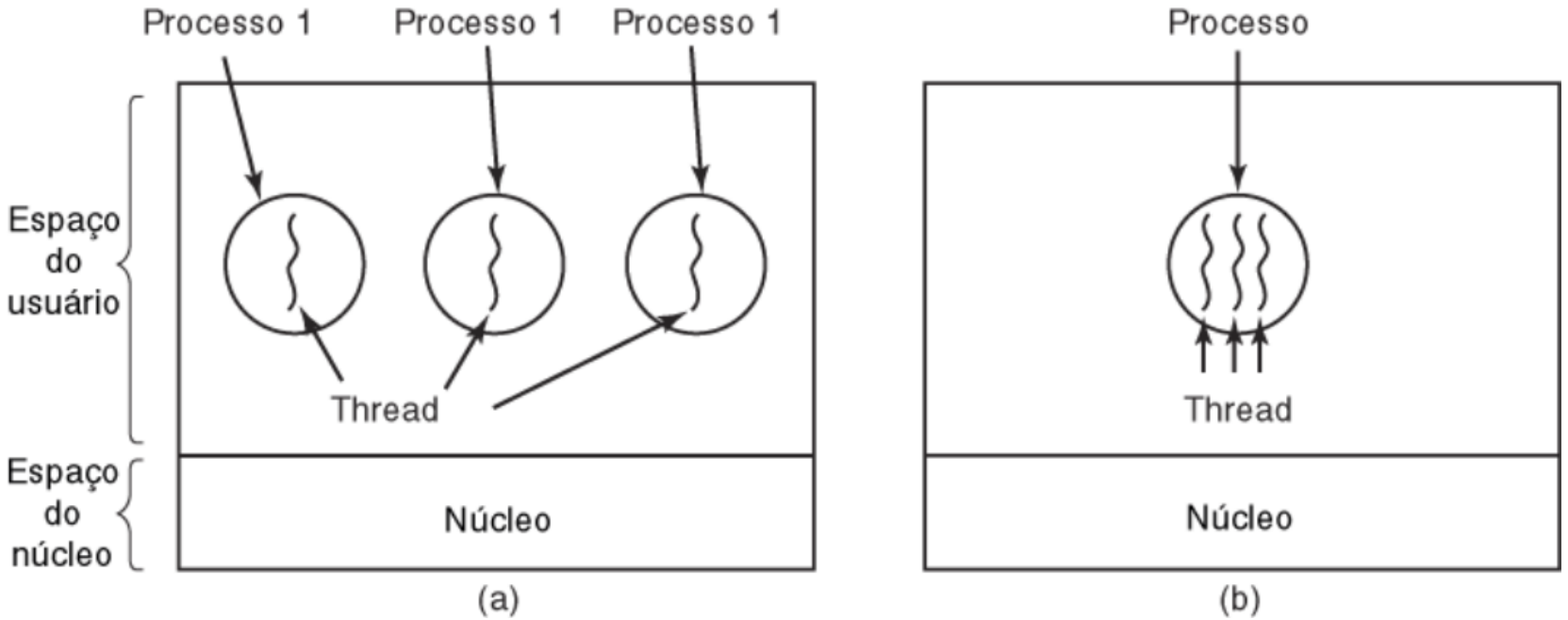


Figura 3: (a) Processo com uma thread. (b) Processo com três threads [3].

Threads: Implementação

- Quais recursos as threads compartilham em um processo?

Compartilhado entre threads

- Espaço de endereçamento.
- Variáveis globais.
- Arquivos abertos.
- Processos filhos.
- Alarmes pendentes.
- Sinais e tratadores de sinais.
- Estatísticas de execução.
- Permissões de acesso.

Exclusivo por thread

- Thread ID
- Contador de programa.
- Registradores.
- Pilha (variáveis locais e endereços de retorno).
- Estado.
- Prioridade.

Threads: Implementação

- Modelos clássicos - Detalhes:

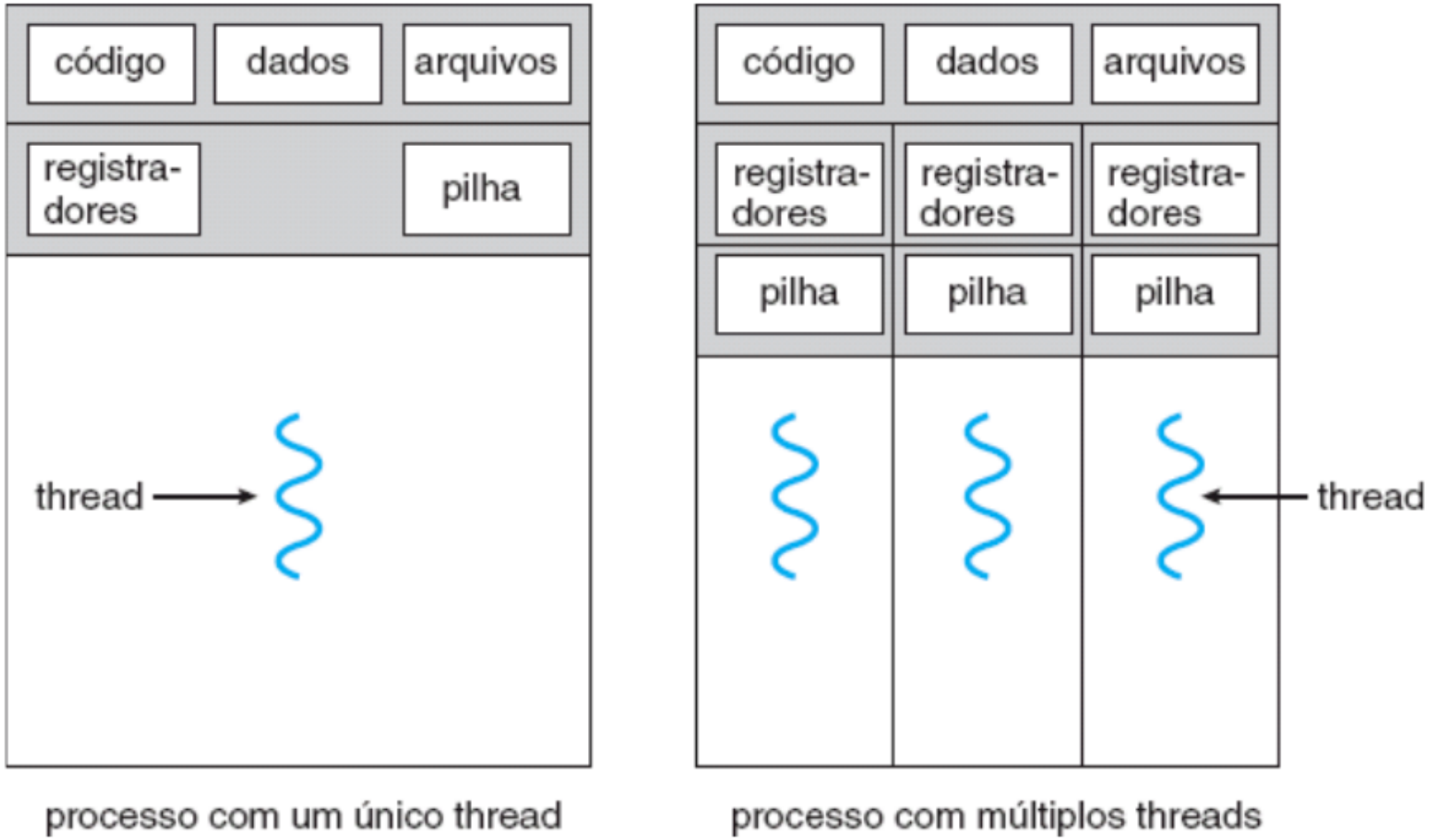


Figura 4: (a) Processo com uma thread. (b) Processo com três threads [2].

Threads: Implementação

Tipos de Implementação de Threads

- Threads do Usuário (*User Level Threads*): Gerenciamento de thread realizado por biblioteca em nível de usuário.
- Threads do Núcleo (*Kernel Level Threads*): Gerenciamento de thread realizado diretamente pelo núcleo do SO.

Threads: Implementação

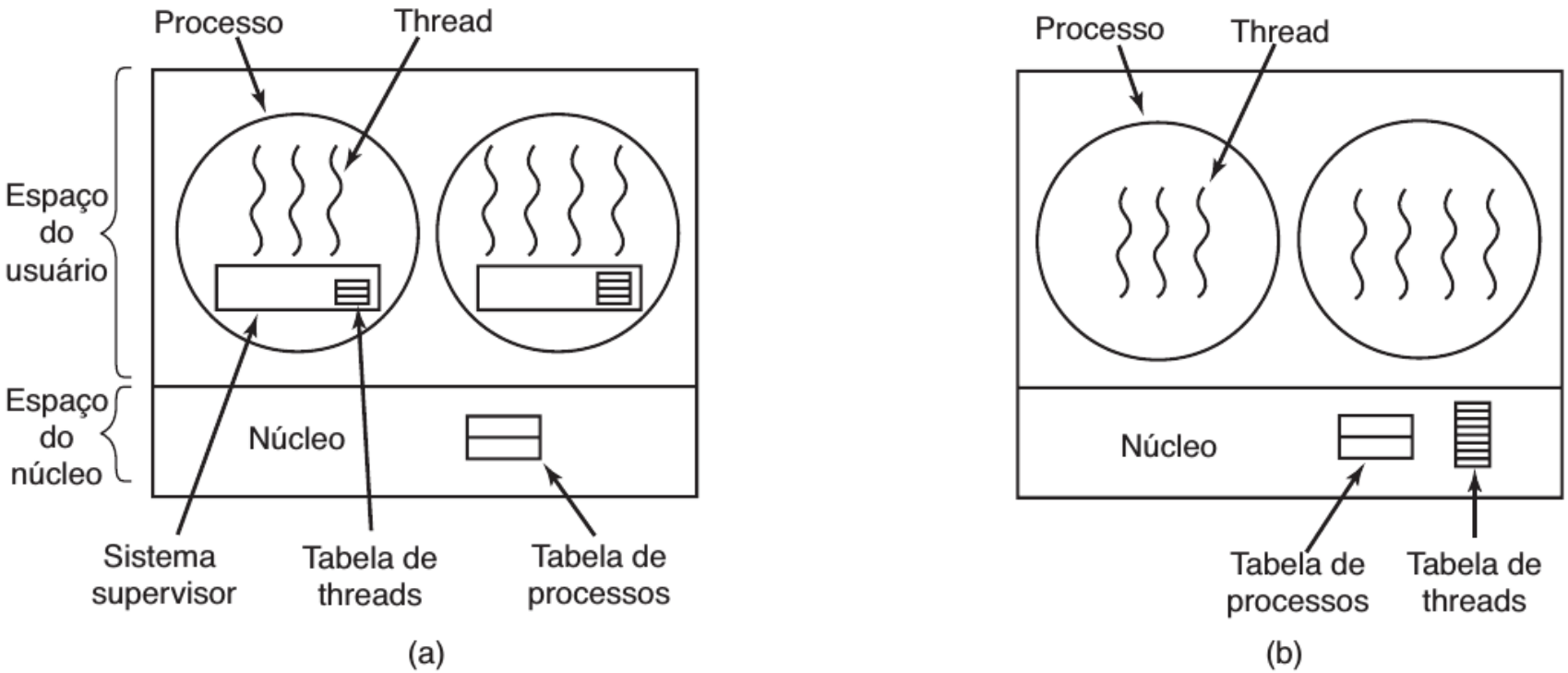


Figura 5: (a) Threads em nível de usário. (b) Threads em nível de núcleo [3].

Threads: Nível de Usuário

- Executam operações de suporte a threads no espaço do usuário.
- SO mapeia threads de um processo multithread para um único contexto de execução.
- Vantagens:
  - Escalonamento controlado pela biblioteca possibilita otimizar o desempenho.
  - Chaveamento de contexto em nível de usuário.
  - Portabilidade maior.
- Desvantagens:
  - Núcleo considera o processo multithread como um único thread.
  - Operações de E/S podem bloquear todas as threads.
  - Escalonamento limitado a um único núcleo.

Threads: Nível de Núcleo

- Mapeia cada thread para o seu próprio contexto de execução.
- O núcleo reconhece cada thread individualmente.
- Vantagens:
  - paralelismo.
  - divisão justa do processador.
- Desvantagens:
  - custo do chaveamento de contexto.
  - menor portabilidade.

Thread: Implementação Híbrida

- Consiste em mapear números arbitrários de threads de usuário para threads de núcleo.

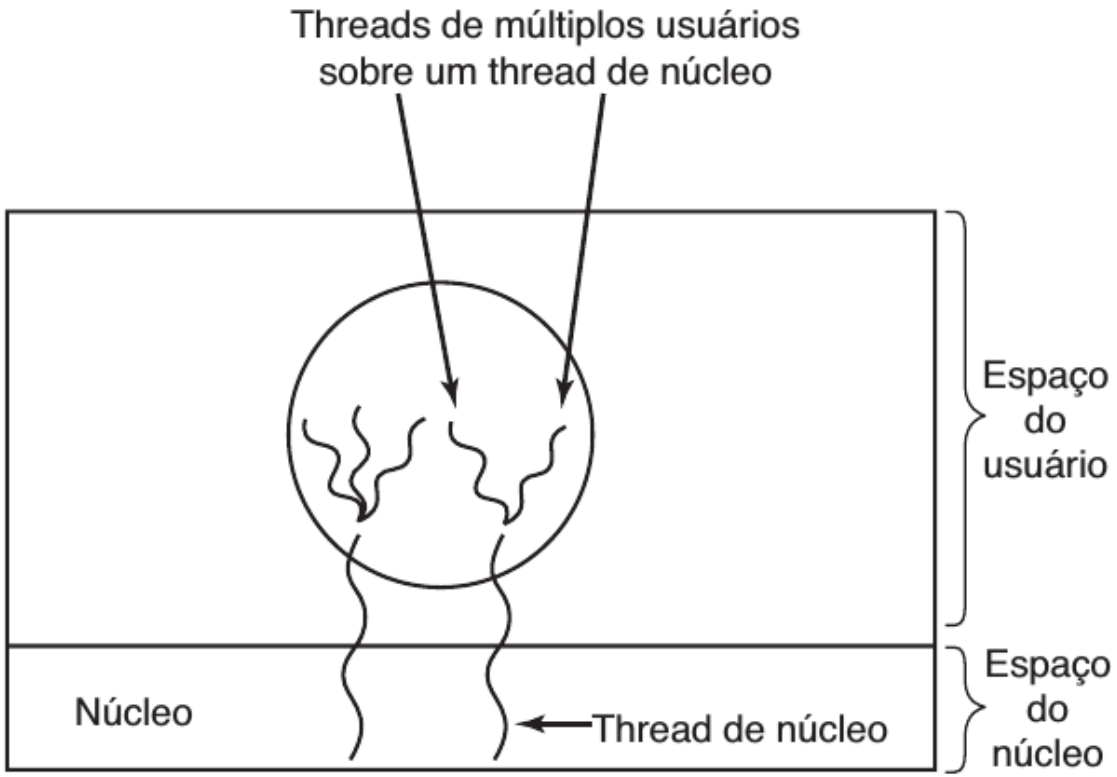


Figura 6: Threads de um processo mapeadas em duas threads de núcleo [3].

Threads: Implementação Híbrida

- Necessitam de comunicação para manter o número apropriado de threads de núcleo alocados à aplicação.
- O mecanismo de **ativações do escalonador** oferece **upcalls** – um mecanismo de comunicação do núcleo para a biblioteca de threads.

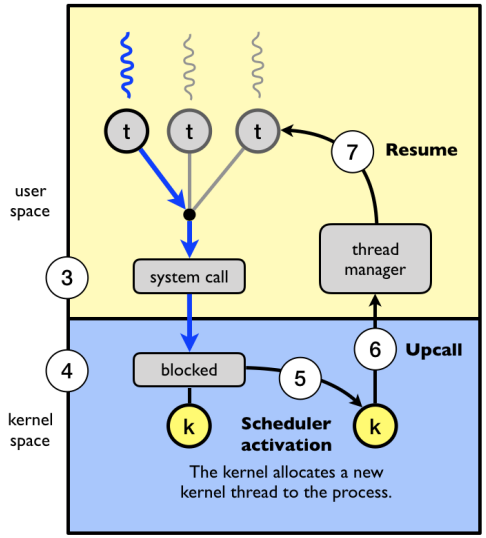
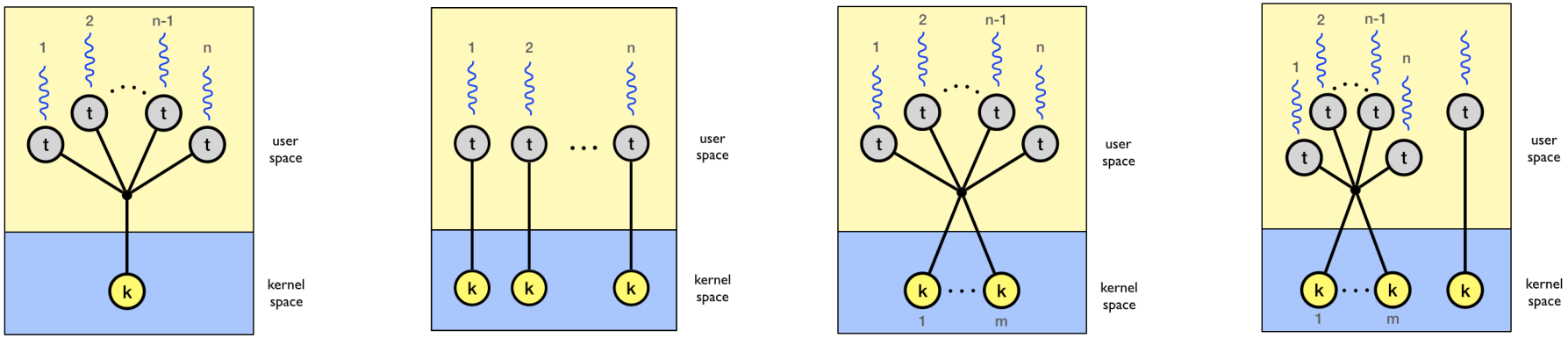


Figura 7: Ativação de escalonador via upcalls.[1]

[1] http://www.it.uu.se/education/course/homepage/os/vt18/module-4/implementing-threads/

Threads: Modelos de mapeamento

- Muitos-para-um: Solaris Green Threads, GNU Portable Threads.
- Um-para-um: Windows NT/XP/2000, Linux, Solaris 9 em diante
- Muitos-para-muitos (dois níveis): Solaris <= 8, Windows NT/2000 com *fiber* .



Fonte: `http://www.it.uu.se/education/course/homepage/os/vt18/module-4/implementing-threads/`

Threads: Tratamento de sinais

- Sinais são usados em sistemas UNIX para notificar um processo de que ocorreu um evento em particular;
- Opções para tratamento de sinais com threads:
  - Entregar o sinal a thread de destino no processo.
  - Entregar o sinal a todas threads no processo.
  - Entregar o sinal a threads específicas no processo.
  - Atribuir uma área específica para receber todos os sinais para o processo.
- As threads normalmente estão associadas a um conjunto de sinais pendentes que são emitidos quando são executadas.
- As threads podem mascarar todos os sinais, exceto aqueles que deseja receber.

Threads: Operações

Operações com threads

- criação e finalização.
- sincronização (*join, blocking*).
- escalonamento (priority, yield).
- sinalização (síncrona ou assíncrona - processo externo).
- cancelamento (assíncrono, adiado).
- suspensão (parada) e ativação (retomada).
- reuso (*pools de thread*)

Atividades

1. Fazer a lista de exercícios *L03 - Threads* disponível na plataforma Moodle.
2. Sugestões de leitura:
   - Threads (Seção 2.2). Sistemas Operacionais Modernos, Tanenbaum and Bos [3].
   - Threads (Seção 5.4). Sistemas Operacionais: Conceitos e Mecanismos, Maziero [1].

Referências I

[1] Maziero, C. A. (2017). *Sistemas operacionais: conceitos e mecanismos*. online. Disponível em http://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=so:so-livro.pdf.

[2] Silberschatz, A., Galvin, P. B., and Gagne, G. (2015). *Fundamentos de sistemas operacionais*. LTC, 9 edition.

[3] Tanenbaum, A. S. and Bos, H. (2016). *Sistemas operacionais modernos*. Pearson Education do Brasil, 4 edition.