

# ***Threads***

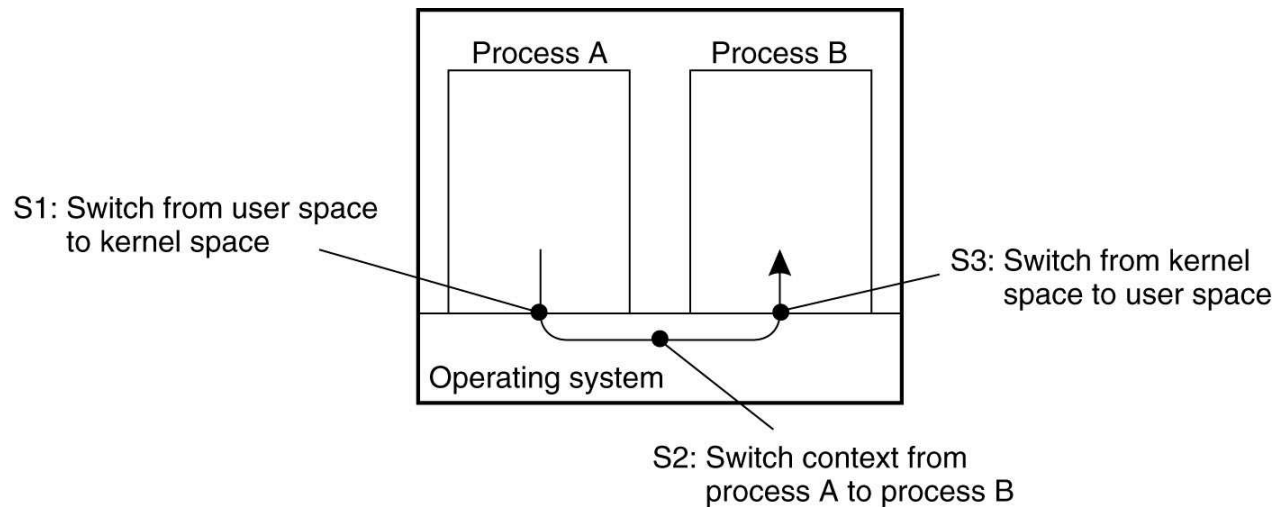
---

- Introdução
- Processos e *Threads*
- Níveis de implementação
- Modelos de *Multithreads*
- Exemplo de uso

# Multiprogramação pesada

---

- Custo de gerenciamento de processos – fator limitante
  - Criação do processo
  - Troca de contexto
  - Esquemas de proteção, memória virtual, ...
- Solução:
  - Aliviar os custos
  - Reduzir o overhead envolvido

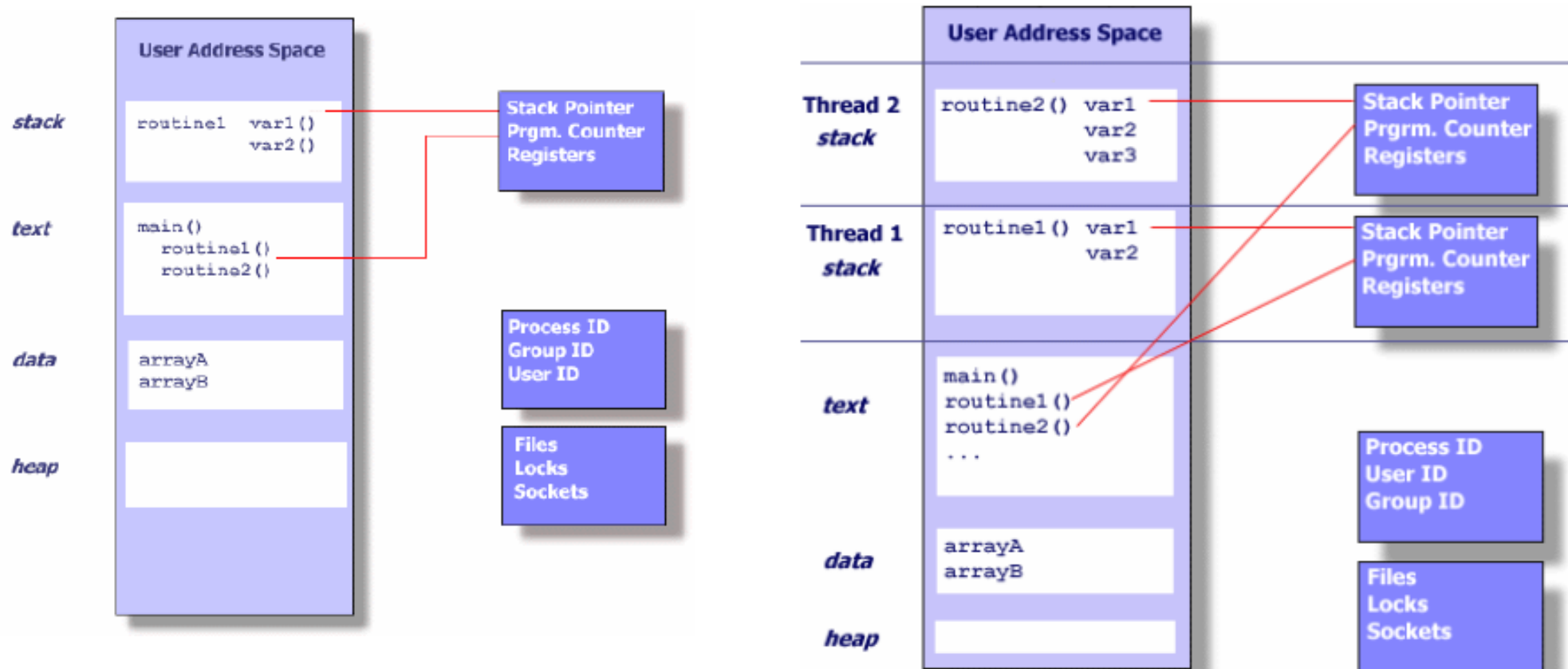


# Multiprogramação leve - Threads

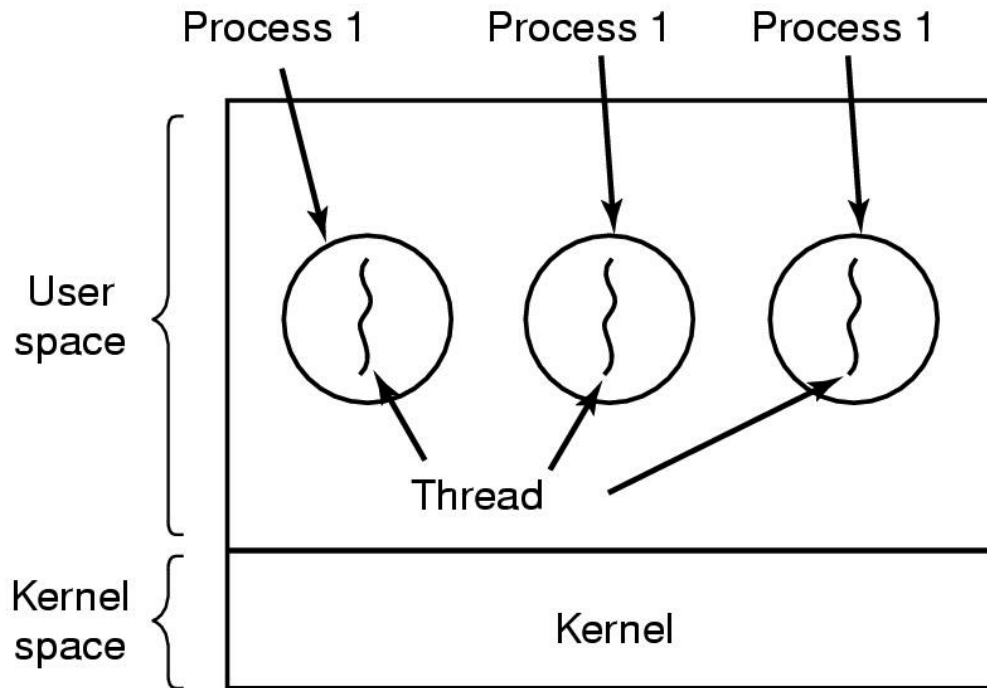
---

- Fornecido pela abstração de um fluxo de execução → **thread**
  - mecanismo que permite a um processo ter mais de um fluxo de controle
  - *threads*: compartilham o espaço de endereçamento (“processo leve”)
  - estados fundamentais: executando, pronta, bloqueada
  - unidade de interação passa a ser a função
  - contexto: pilha, PC, registradores de uso geral
  - comunicação via compartilhamento direto da área de dados
-

# Processos x *Threads* (1)



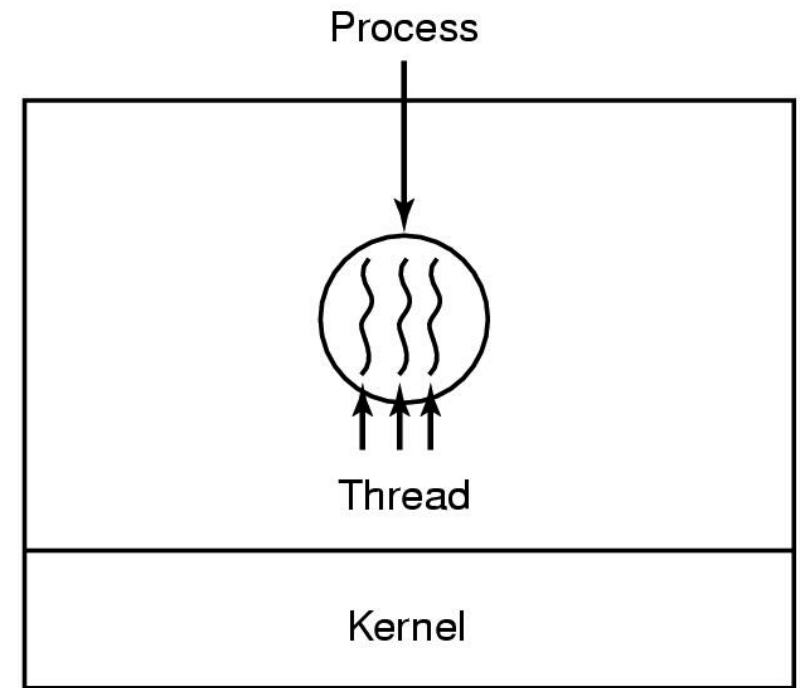
# Processos x *Threads* (2)



(a)

*(a) 3 processos, cada um com uma thread*

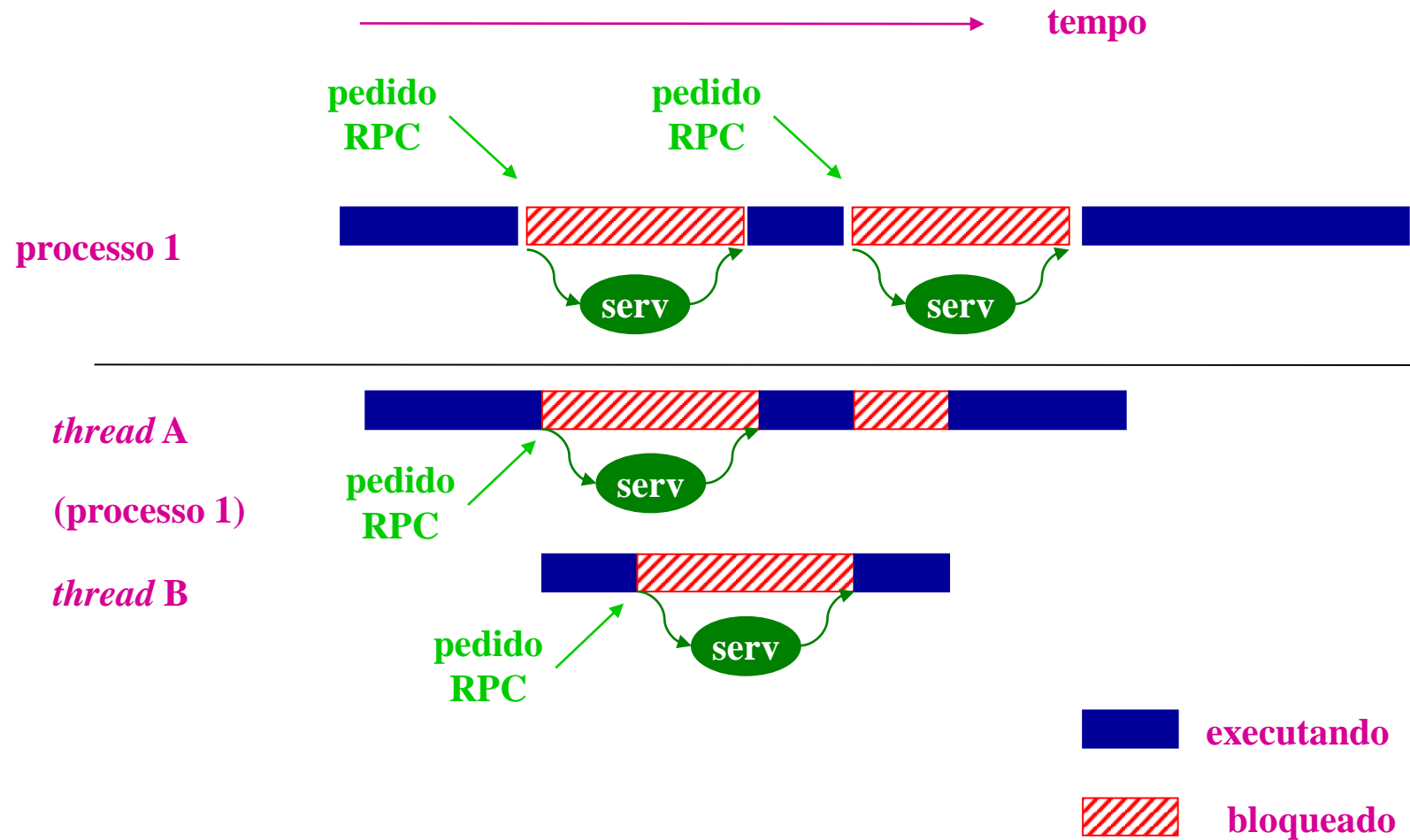
*(b) 1 processo com 3 Threads*



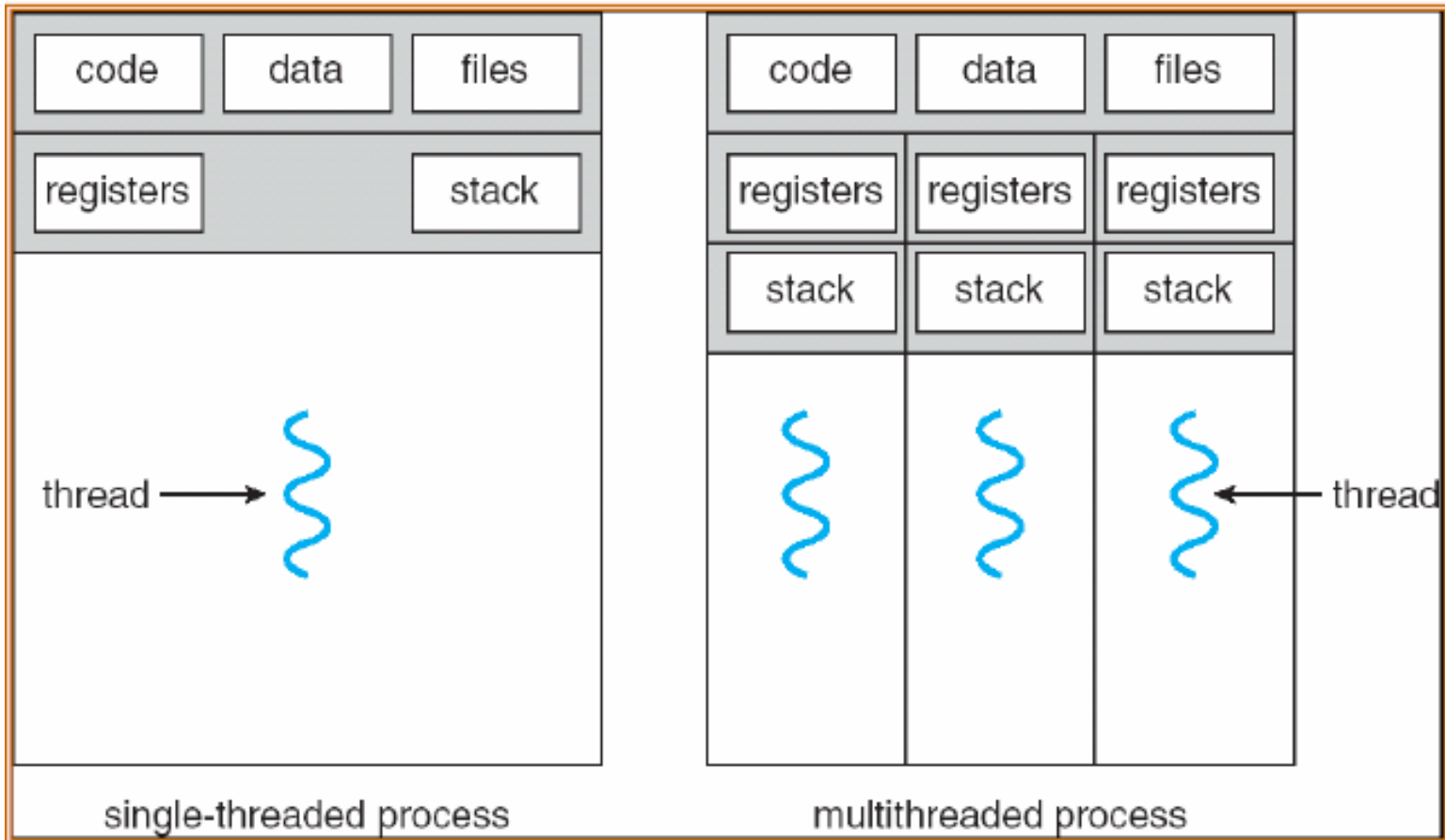
(b)

*Processo* = fluxo de controle +  
espaço de endereçamento  
*Thread* = fluxo de controle

# Exemplo: RPC

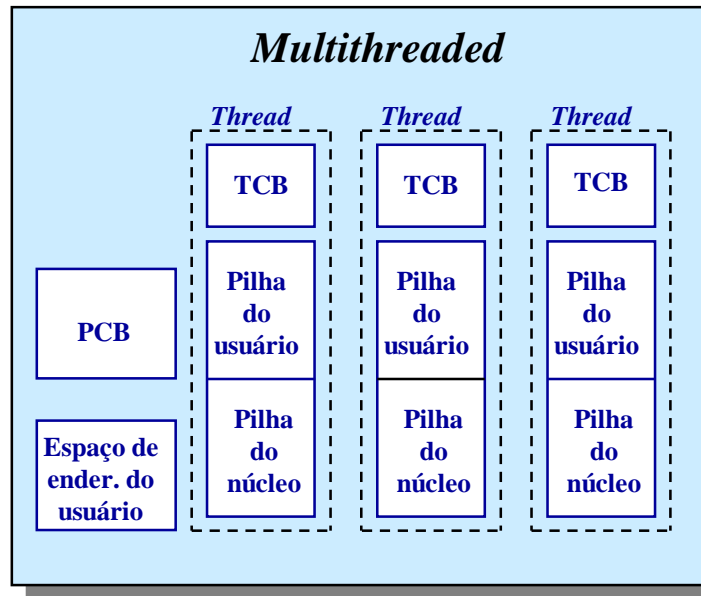
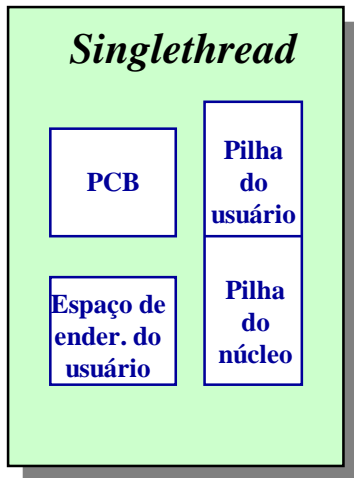


# Single x Multithreaded (1)



# Single x Multithreaded (2)

---



## Multithreading

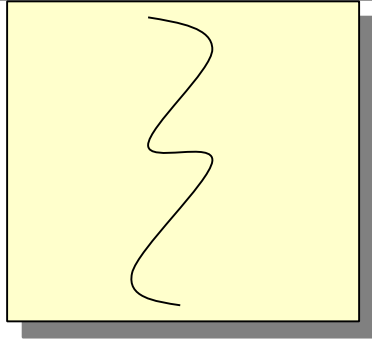
Suporte a múltiplas threads de execução dentro de um único processo

- estruturas de dados similares ao descritor de processo (PCB): TCB



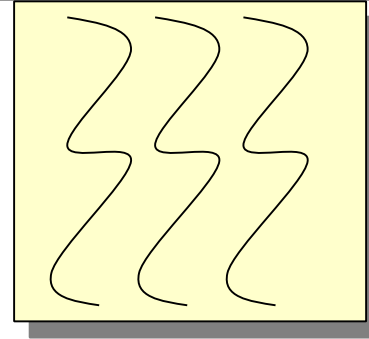
# Single x Multithreaded (3)

Monoprogramação



um processo  
uma thread

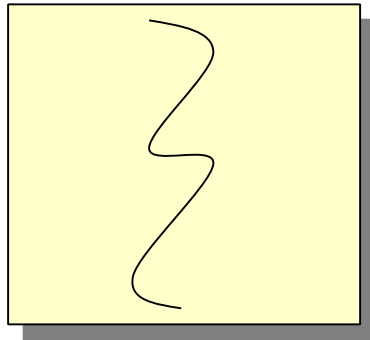
Ex.: MSDOS



um processo  
várias threads

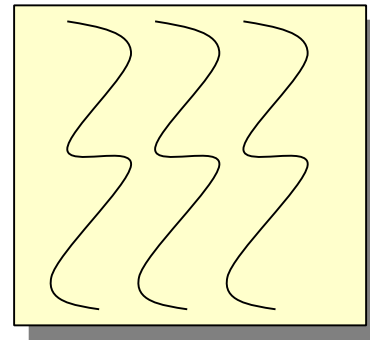
Ex.: JVM

Multiprogramação



vários processos  
uma thread por processo

Ex.: Unix (início)



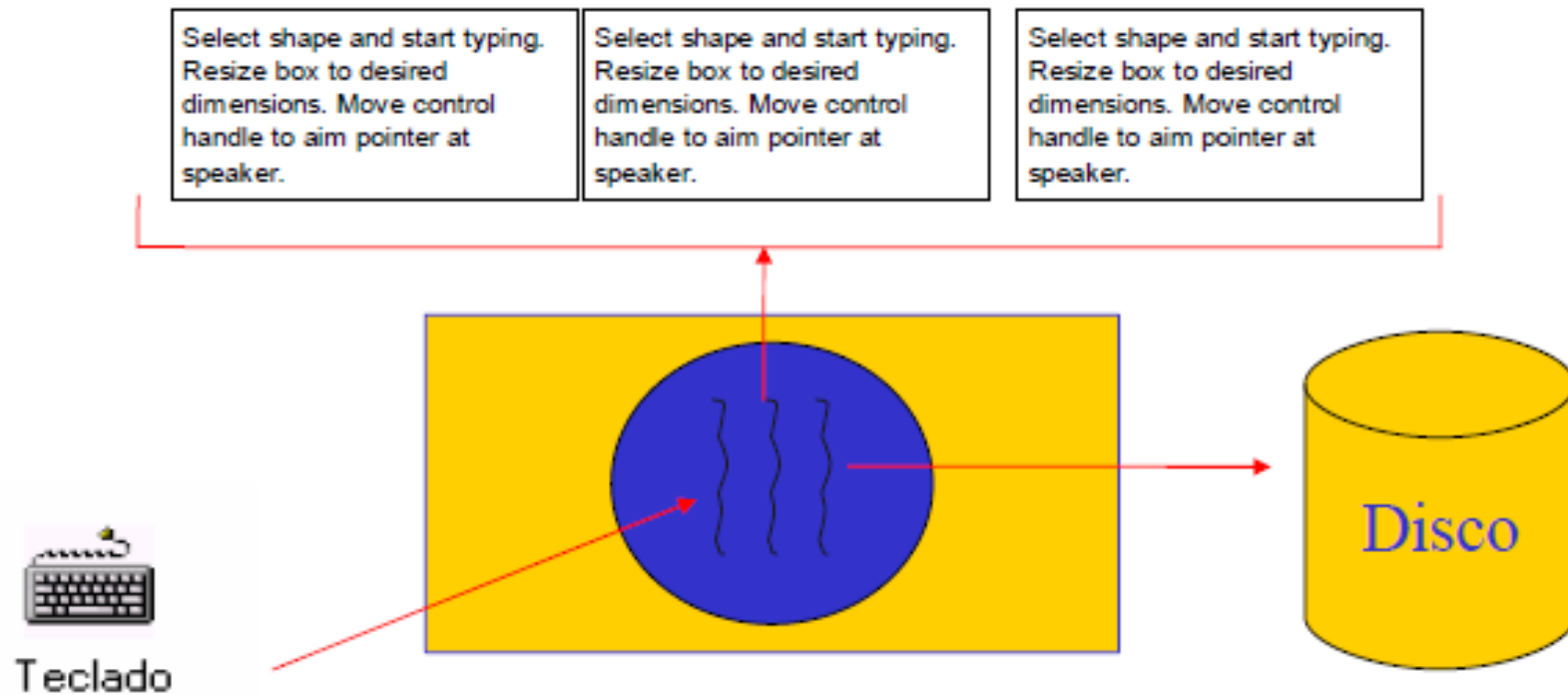
vários processos  
várias threads por processo

Ex.: Linux, Win

# Exemplo de uso de threads (1)

---

- Editor de textos com 3 threads (threads para diferentes tarefas)



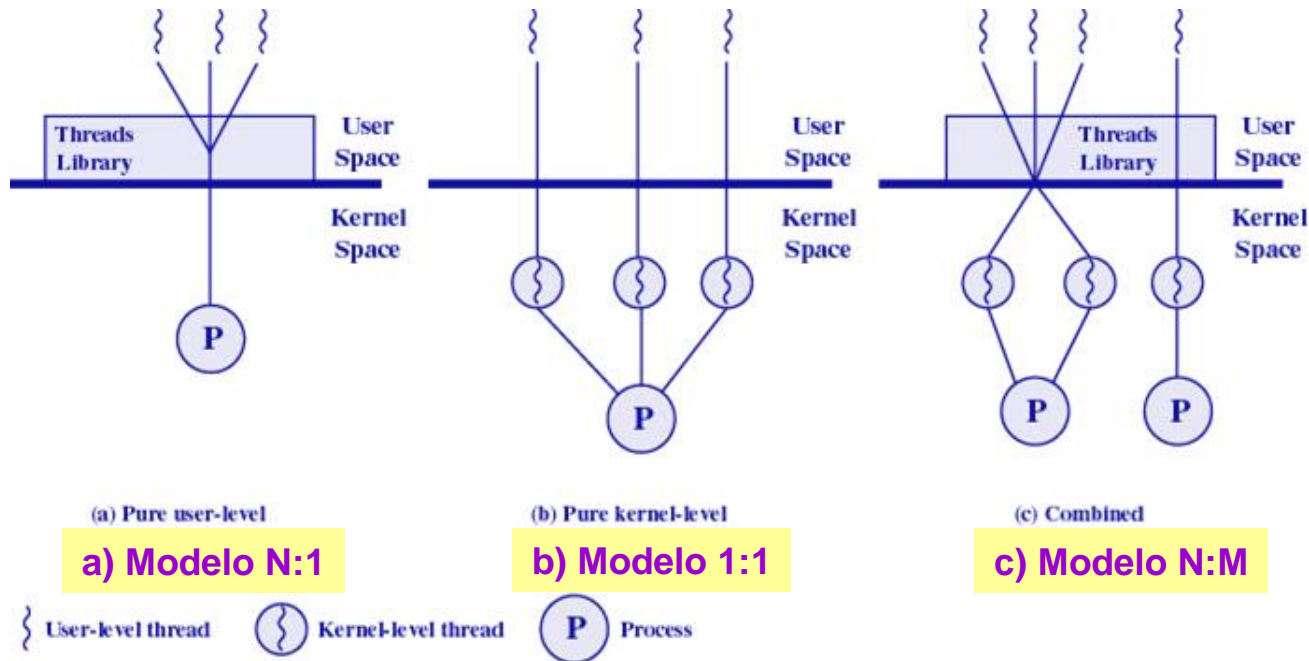
# Exemplo de uso de threads (2)

---

- Processador de textos
  - thread para exibir gráfico
  - thread para ler sequências de teclas digitadas
  - thread para efetuar a verificação ortográfica em segundo plano
- Sistemas computacionais
  - 3 processos separados não funcionam – os 3 precisam atuar sobre o mesmo documento
  - 3 threads compartilham uma memória comum, permitindo acesso ao documento

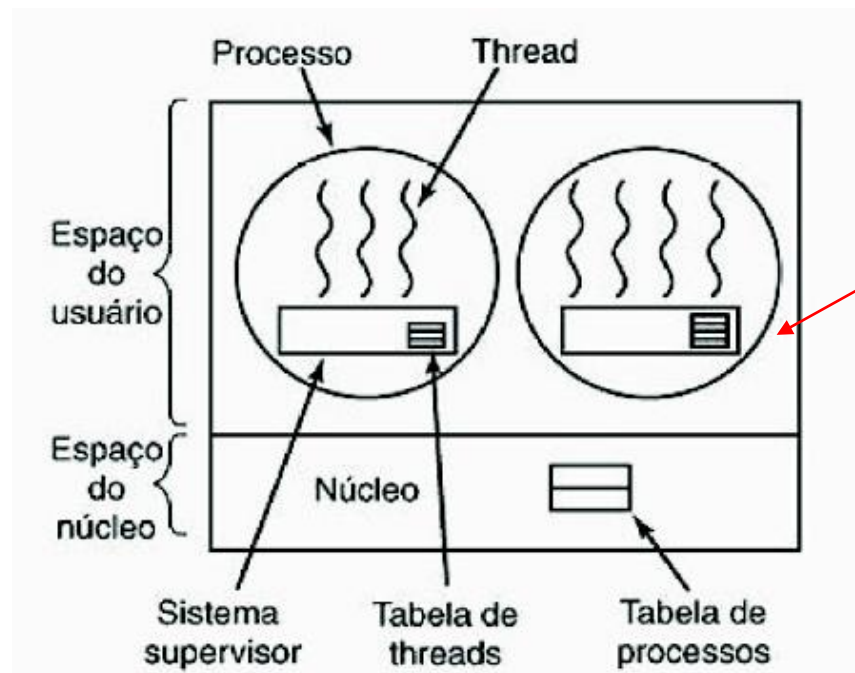
# Thread: implementação

- 2 principais níveis de implementação:
  - Threads no espaço de usuário (user level threads, N:1)
  - Threads no espaço de kernel (kernel level threads, 1:1)



# Threads de usuário

- admitidas no nível do usuário e gerenciadas sem o suporte do kernel
  - thread requisita I/O → bloqueia todo o processo



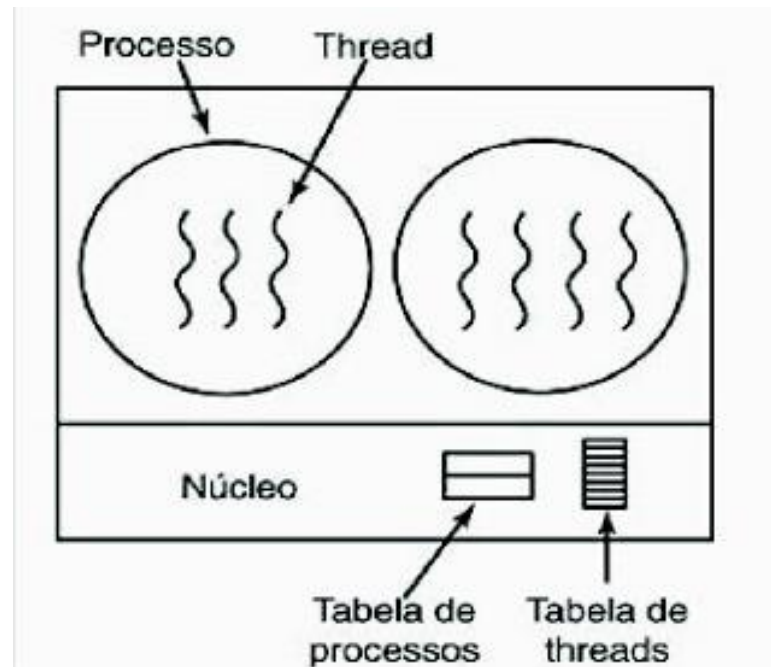
Biblioteca de threads com:

- Tabela de threads
- Escalonador próprio

# Threads de kernel

---

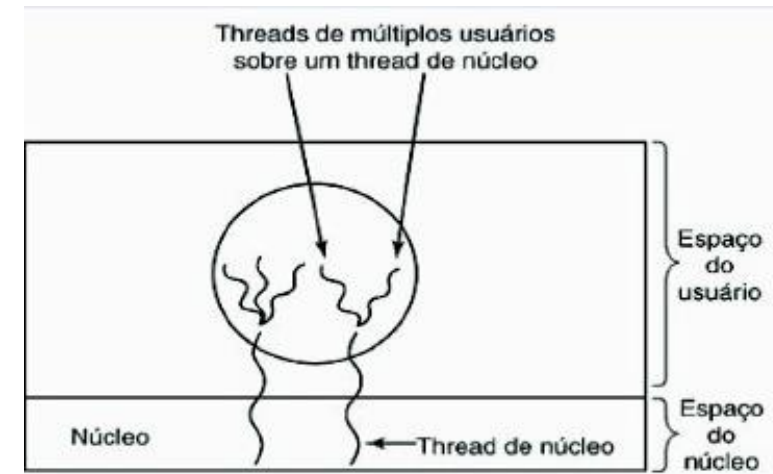
- admitidas e gerenciadas diretamente pelo SO
  - kernel chaveia entre threads, independente do processo a que pertencem



# Threads: implementação híbrida

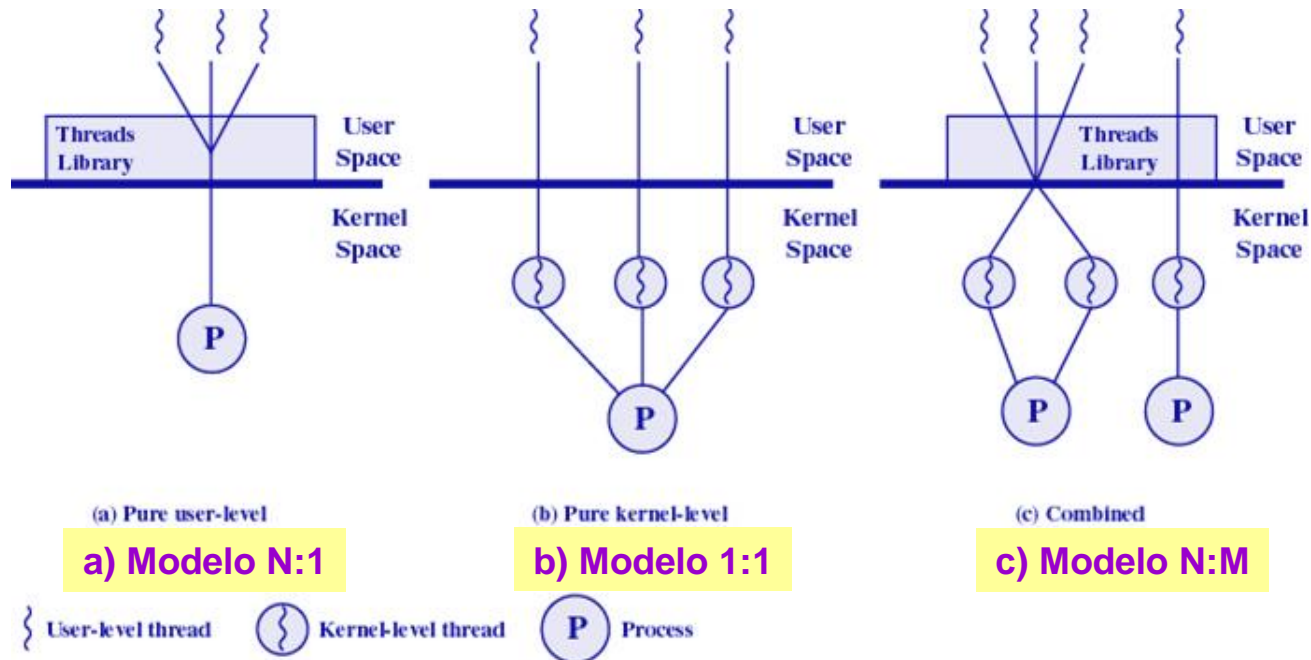
---

- Tenta combinar as vantagens dos 2 modos anteriores
  - Usuário: rápida criação e chaveamento entre threads
  - Kernel: o processo todo não é bloqueado pelo bloqueio de uma thread
- A ideia é utilizar algumas threads de kernel e multiplexar threads de usuário sobre elas



# Modelos de multithreading

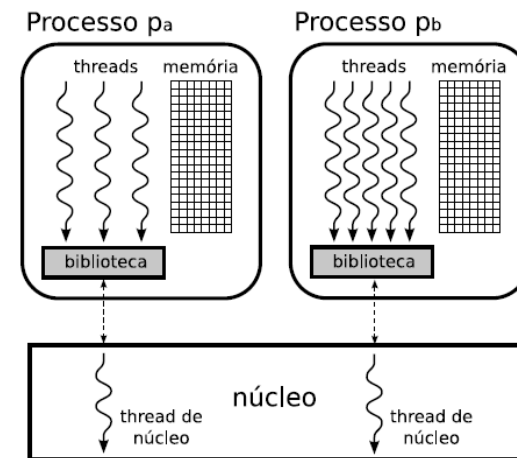
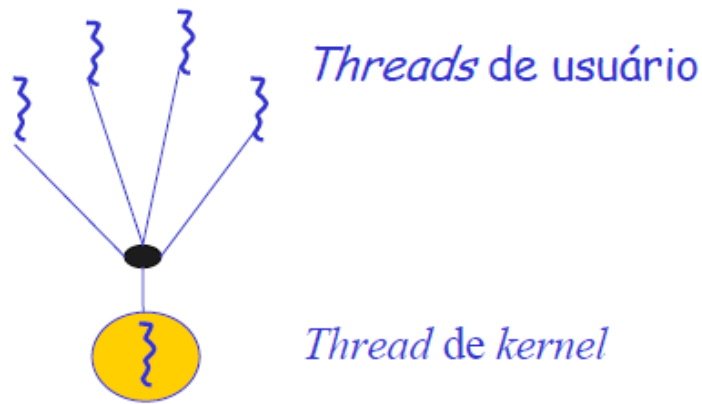
---



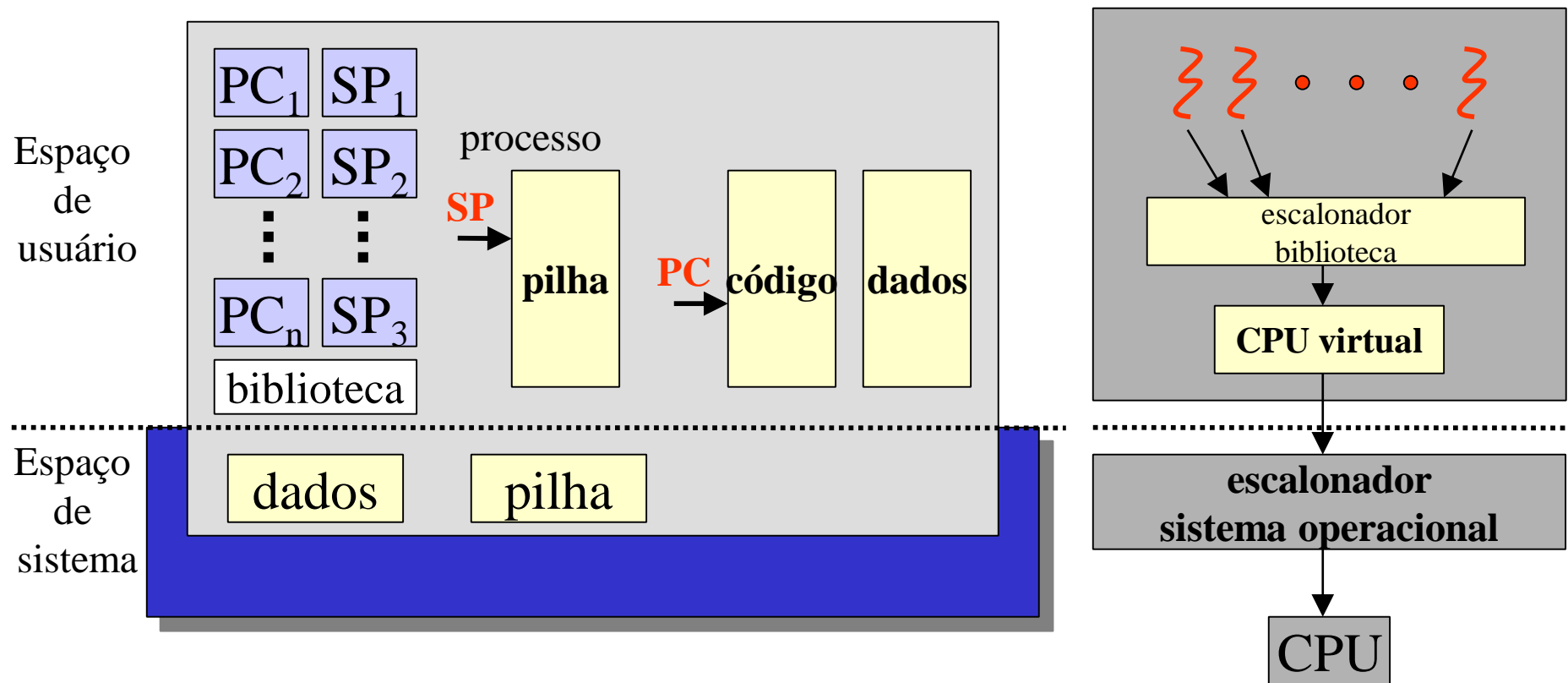


## a) Modelo N:1, user-level threads (1)

- **N threads** de usuário mapeadas para **1 thread** de kernel
- Gerenciamento feito pela biblioteca de threads no nível de usuário
- Se uma thread faz chamada de sistema bloqueante, todo o processo será bloqueado
- Ex.: GNU Portable threads, POSIX Pthreads, SunOS

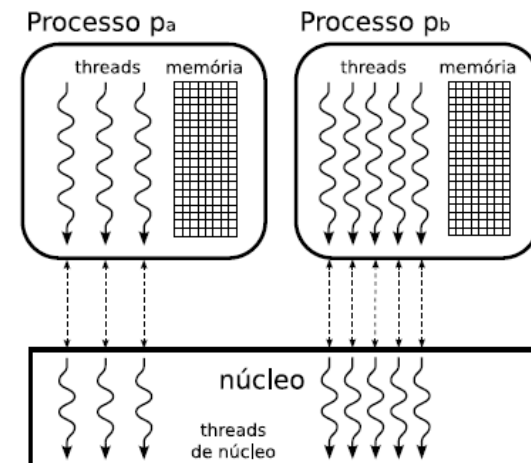
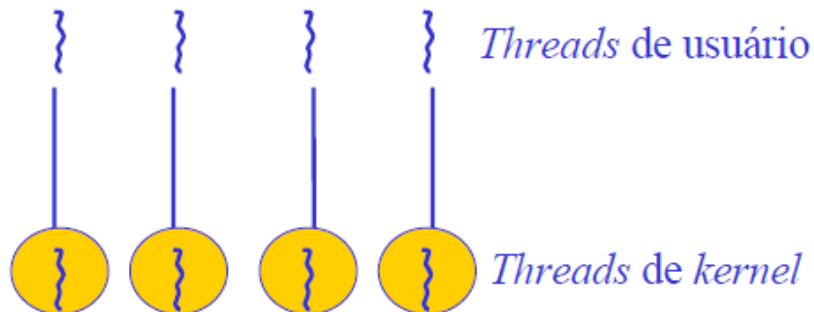


## a) Modelo N:1, user-level threads (2)

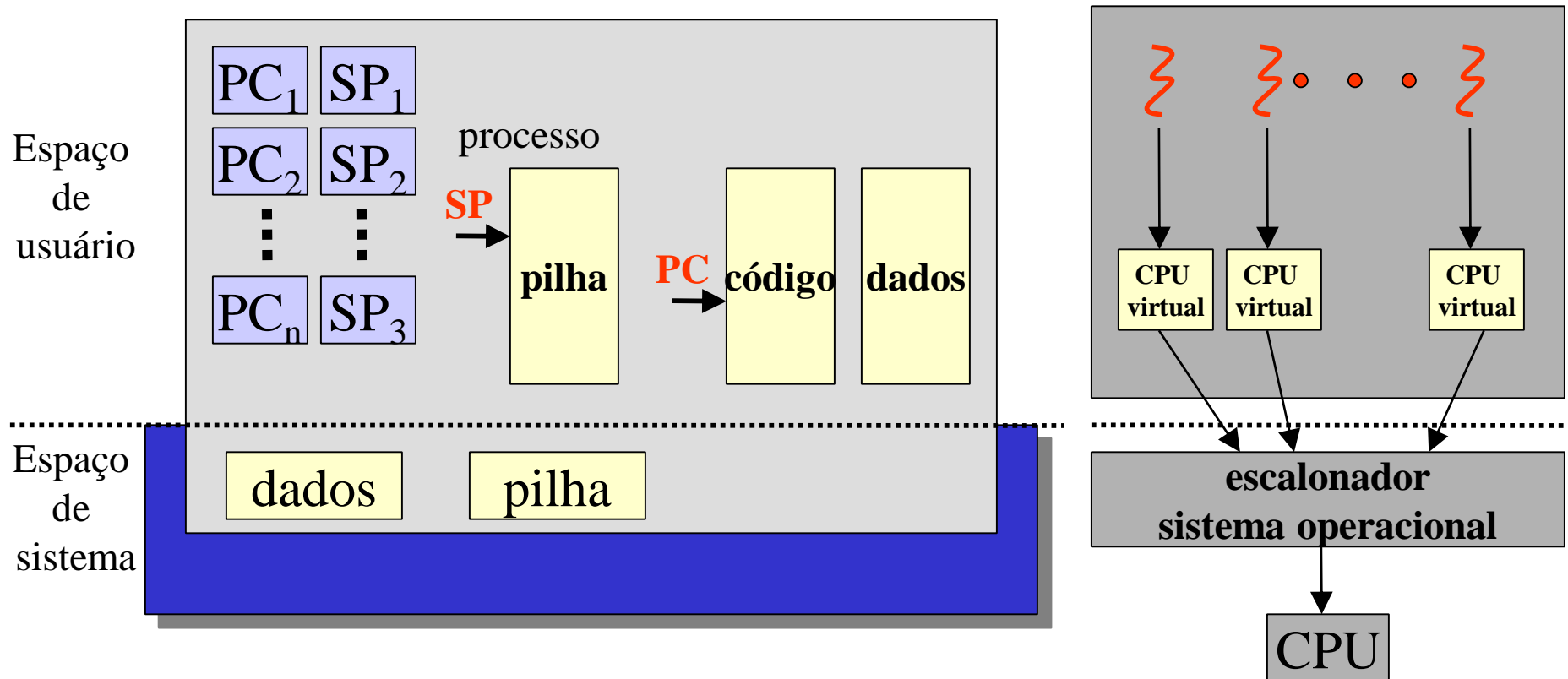


## b) Modelo 1:1, kernel-level threads (1)

- Mapeia **cada thread** de usuário para **1 thread** de kernel
- Thread é a unidade de escalonamento do núcleo
- Biblioteca de chamadas de sistema inclui operações para criar/controlar threads
- SOs que suportam esse modelo: Linux Threads, Win32, FreeBSD

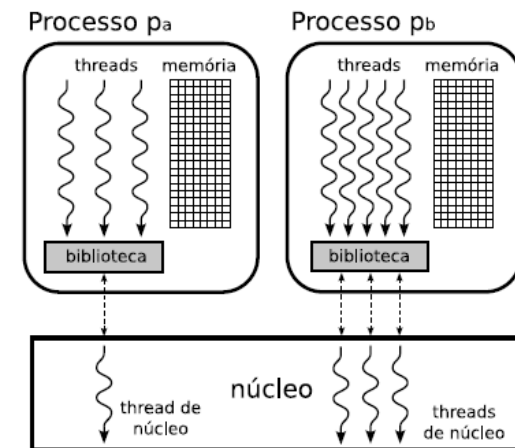
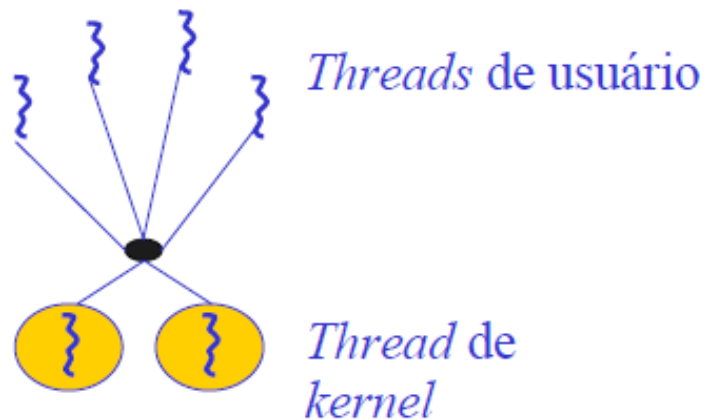


## b) Modelo 1:1, kernel-level threads (2)

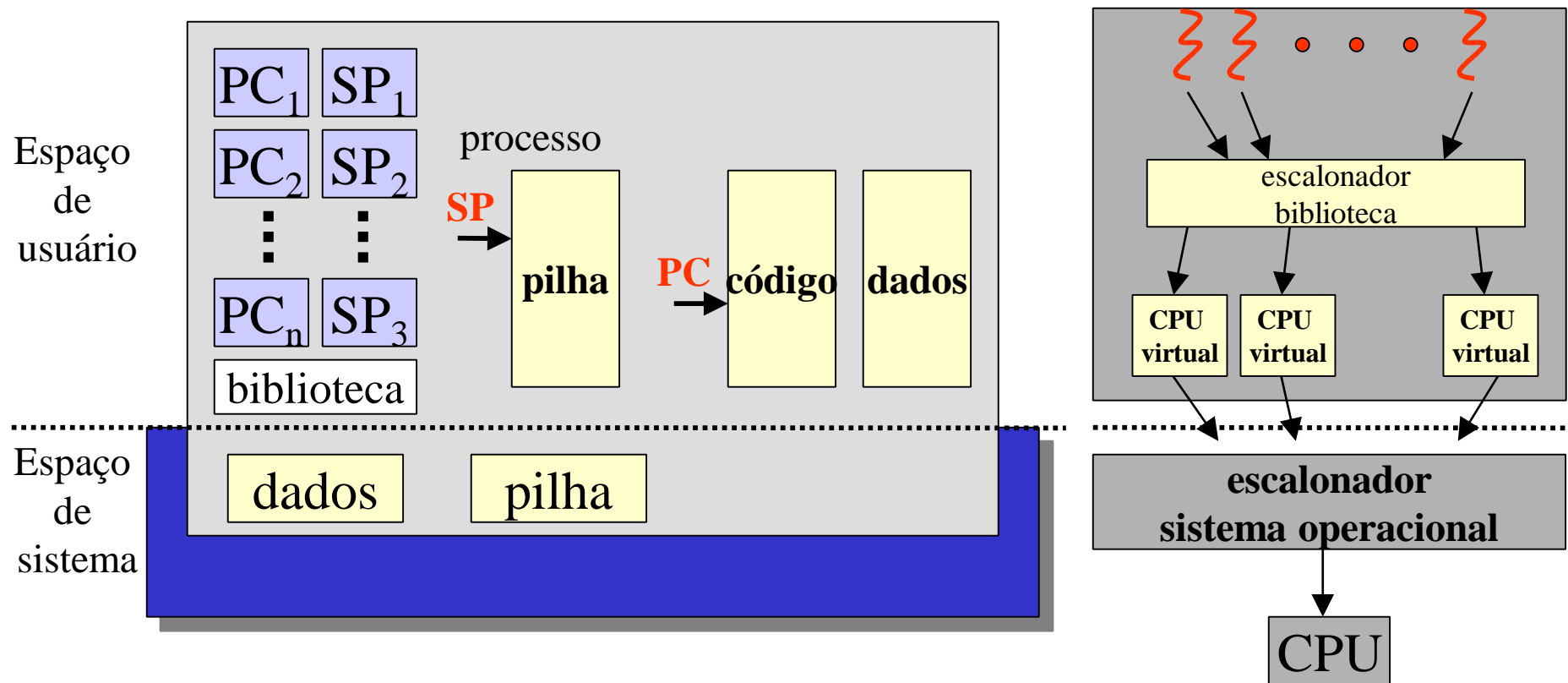


## c) Modelo N:M, híbrido (1)

- Mapeia **N threads** de usuário para **M threads** de kernel ( $M < N$ )
- Quando uma thread faz chamada de sistema bloqueante, SO pode escalonar outra thread do mesmo processo
- SOs que suportam esse modelo: Win7, Solaris 8



## c) Modelo N:M, híbrido (2)



# Bibliotecas de Threads

---

- Oferece uma API para a criação e gerenciamento de threads
- 2 formas de implementar bibliotecas:
  - Biblioteca no espaço do usuário sem suporte do kernel
  - Biblioteca no espaço do núcleo com suporte direto do SO
- Bibliotecas mais comuns
  - POSIX Pthreads (nível de usuário ou de kernel)
  - Win32 (nível de kernel)
  - Java (nível de usuário, mas usa a biblioteca do SO hospedeiro)

# Exemplo de uso: Pthreads no C (1)

---

```
#include <pthread.h>
#include <stdio.h>

void * Thread0() {
    int i;

    for(i=0;i<10;i++)
        printf(" Thread0 - %d\n",i);
}

void * Thread1() {
    int i;

    for(i=10;i<20;i++)
        printf(" Thread1 - %d\n",i);
}
```

```
int main() {
    pthread_t t0, t1;
    pthread_create(&t0, NULL, Thread0, NULL);
    pthread_create(&t1, NULL, Thread1, NULL);
    pthread_join(t0, NULL);
    pthread_join(t1, NULL);
    printf("Main ....\n");
}
```

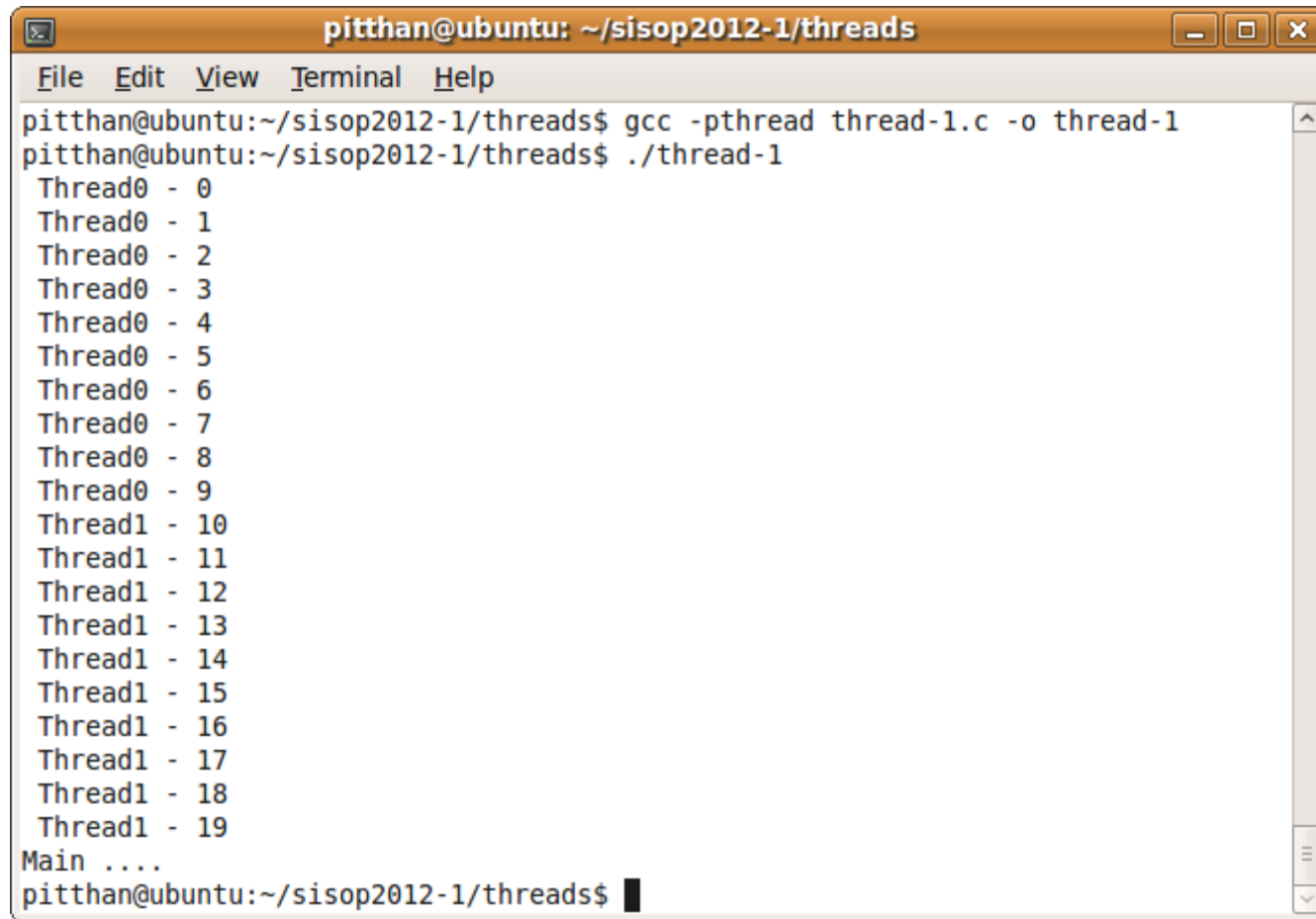
```
// p/ compilar: gcc -o thread thread.c -pthread
```

---



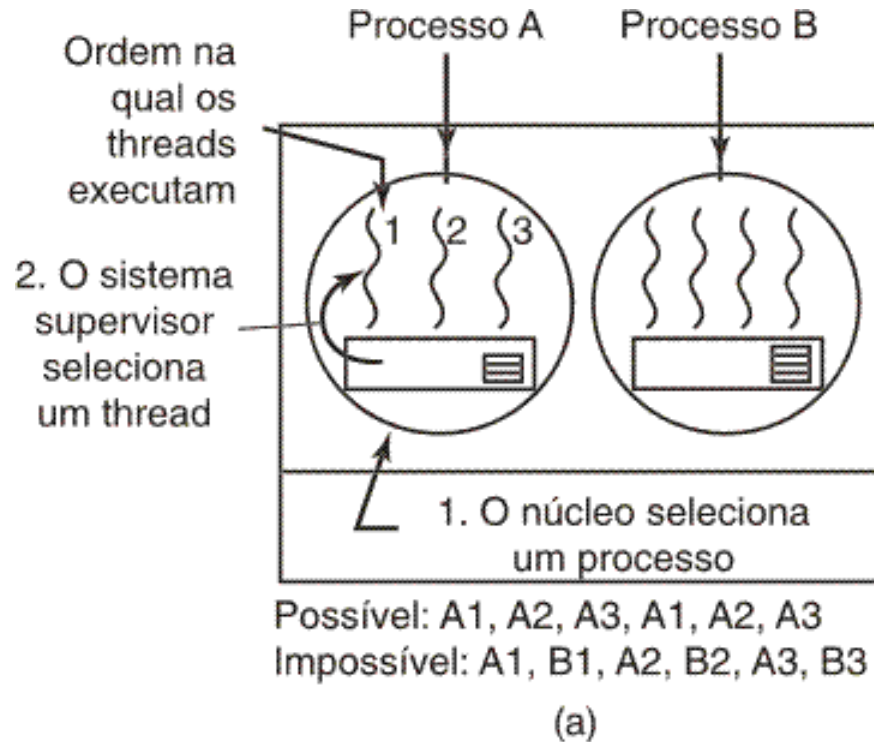
# Exemplo de uso: Pthreads no C (2)

---

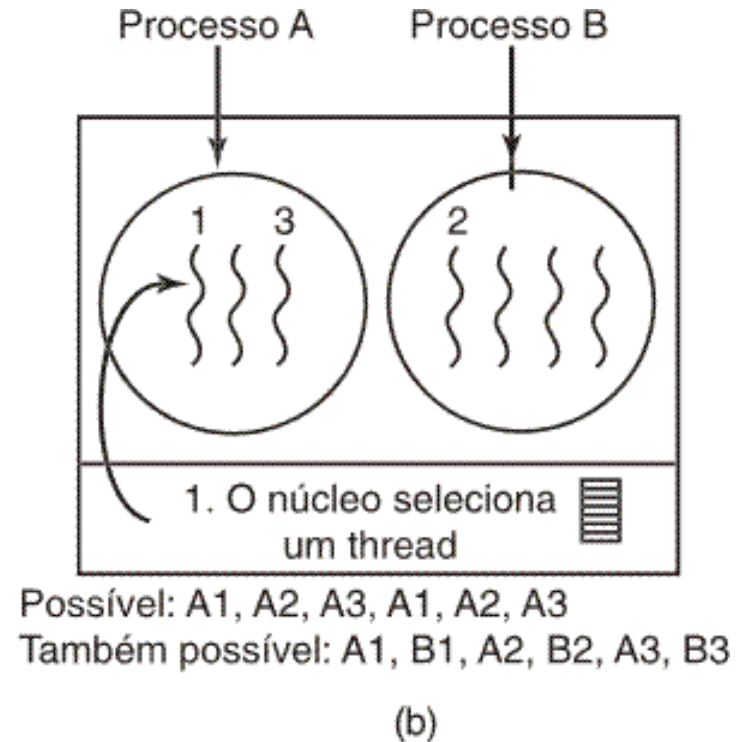
A terminal window titled 'pitthan@ubuntu: ~/sisop2012-1/threads' with a menu bar (File, Edit, View, Terminal, Help). The terminal shows the compilation of 'thread-1.c' using 'gcc -pthread' and the execution of './thread-1'. The output lists thread IDs: Thread0 (0-9) and Thread1 (10-19), followed by 'Main ....' and the shell prompt.

```
pitthan@ubuntu: ~/sisop2012-1/threads
File Edit View Terminal Help
pitthan@ubuntu:~/sisop2012-1/threads$ gcc -pthread thread-1.c -o thread-1
pitthan@ubuntu:~/sisop2012-1/threads$ ./thread-1
Thread0 - 0
Thread0 - 1
Thread0 - 2
Thread0 - 3
Thread0 - 4
Thread0 - 5
Thread0 - 6
Thread0 - 7
Thread0 - 8
Thread0 - 9
Thread1 - 10
Thread1 - 11
Thread1 - 12
Thread1 - 13
Thread1 - 14
Thread1 - 15
Thread1 - 16
Thread1 - 17
Thread1 - 18
Thread1 - 19
Main ....
pitthan@ubuntu:~/sisop2012-1/threads$
```

# Escalonamento de Threads



Threads de usuário



Threads de kernel

# Exemplos de threads

---

- hello.c
- hello\_arg.c
- hello\_args.c
- hello\_join.c