



# HILOS

DEPARTAMENTO DE CIENCIAS E INGENIERÍA DE LA  
COMPUTACIÓN

UNIVERSIDAD NACIONAL DEL SUR

# AGENDA

1. Revisión
2. Clasificación de Hilos
3. Modelos Multihilados
4. Librerías de Hilos
5. Hilos en Linux

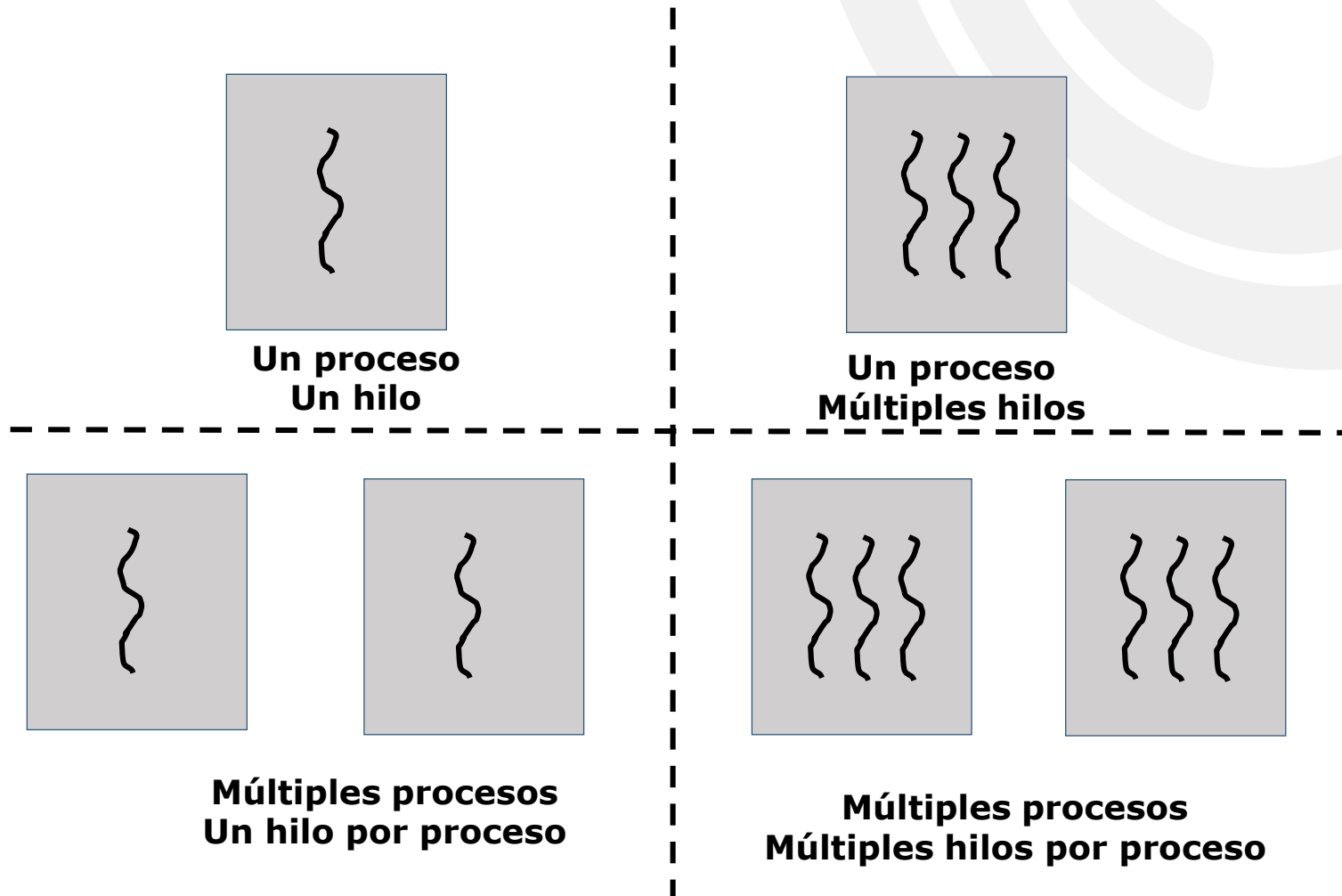
# AGENDA

1. **Revisión**
2. Clasificación de Hilos
3. Modelos Multihilados
4. Librerías de Hilos
5. Hilos en Linux

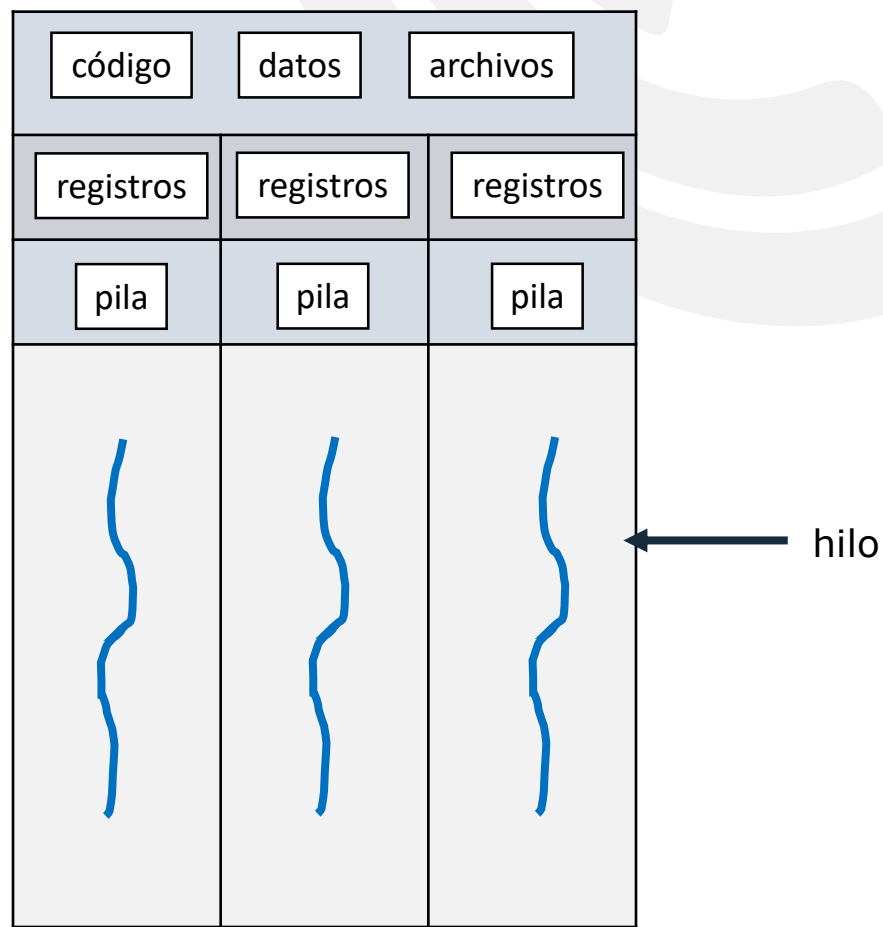
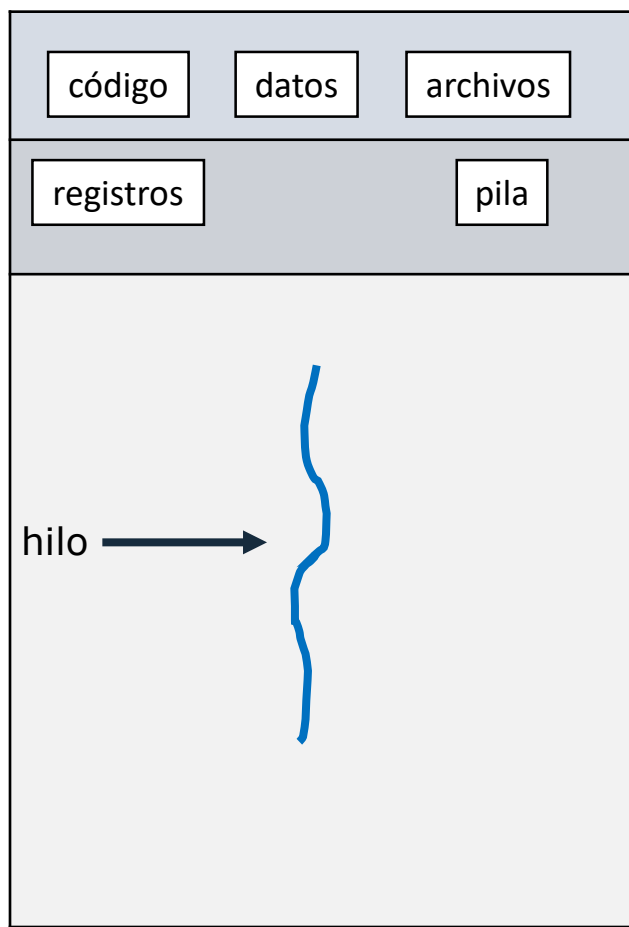
# HILOS

- Un hilo (thread o proceso de peso liviano) es una unidad básica de utilización de CPU, consiste de:
  - contador de programa
  - conjunto de registros
  - espacio de pila
- Un hilo comparte con sus hilos compañeros (colectivamente conocidos como tarea o task) su:
  - sección de código
  - sección de datos
  - recursos del SO.
- Un proceso tradicional o peso pesado es igual a una tarea con un solo hilo.

# HILOS



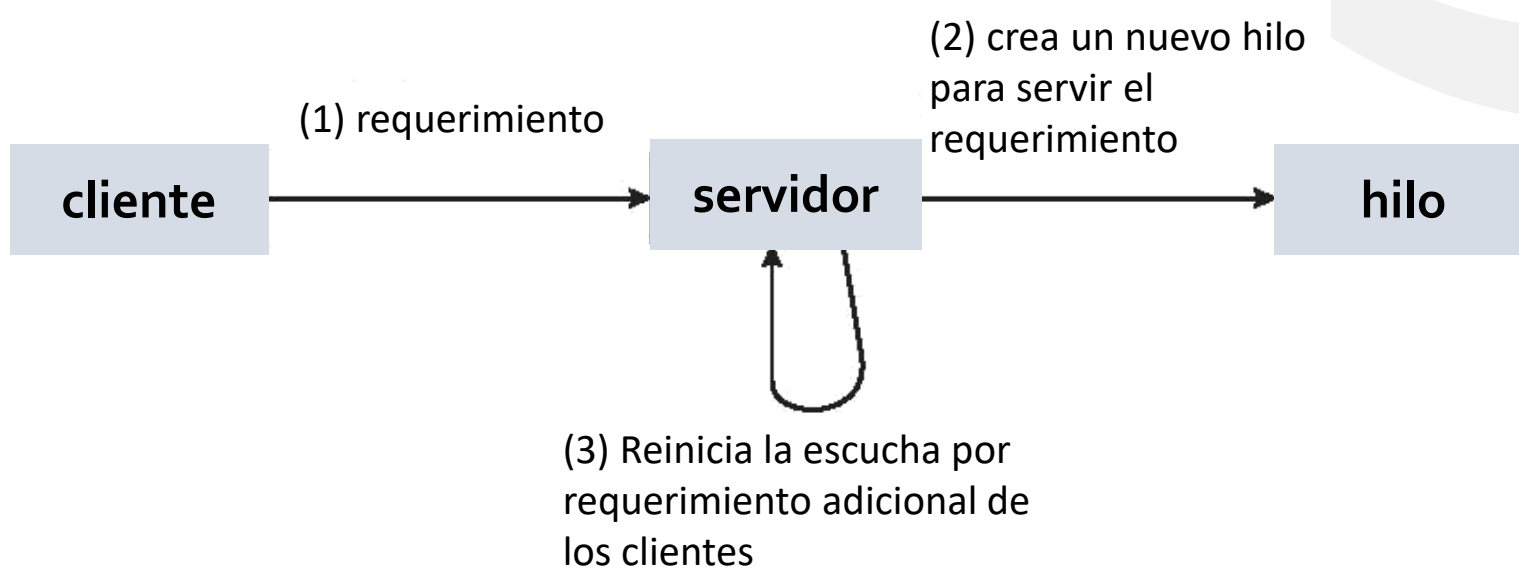
# PROCESOS MONO Y MULTIHILOS



# HILOS

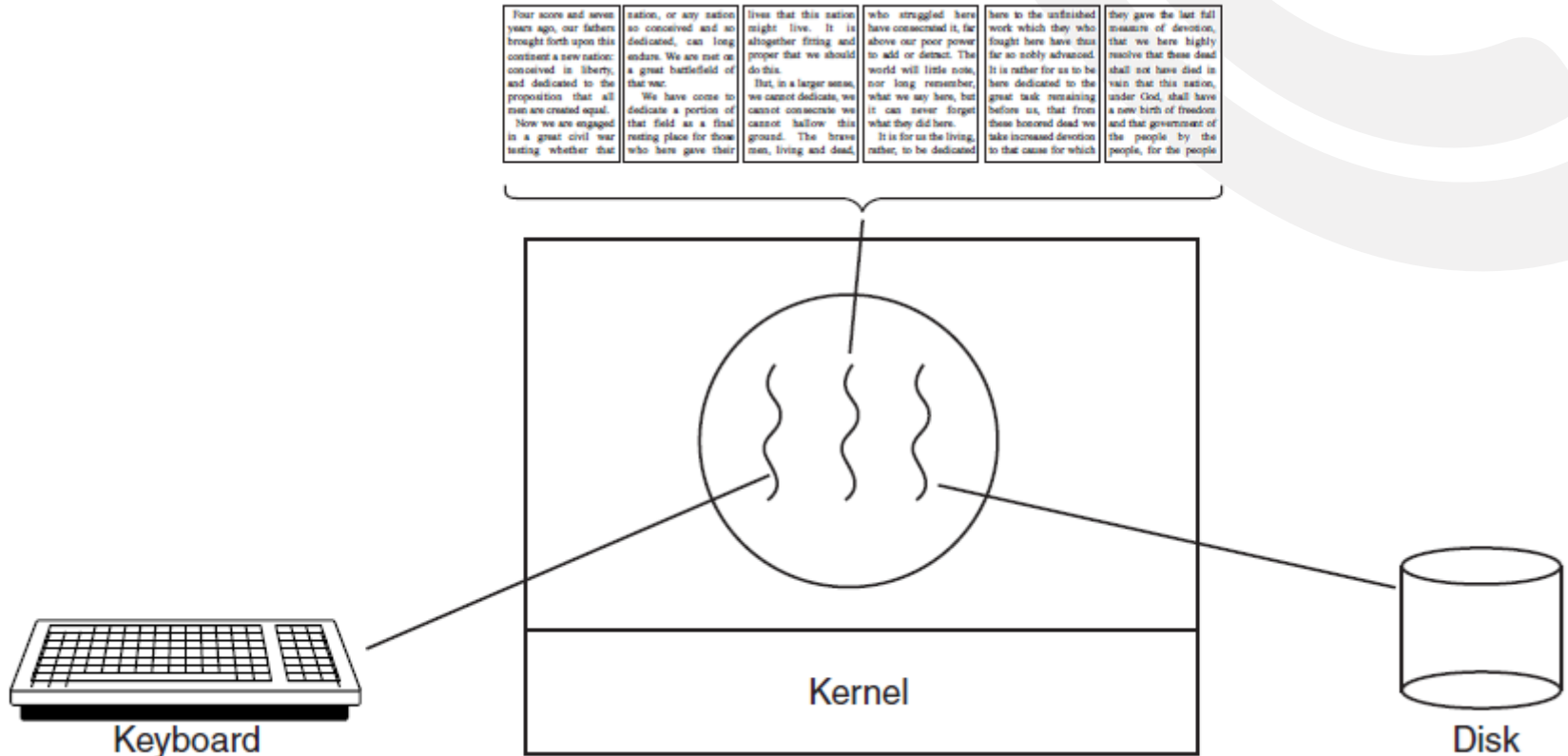
- En una tarea con múltiple hilos, mientras un hilo servidor está bloqueado y esperando, un Segundo hilo de la misma tarea puede estar corriendo.
- Cooperación de múltiple hilos en una misma tarea confiere alto procesamiento total y mejora el rendimiento.
  - Aplicaciones que requieren compartir un buffer común (p.e., productor-consumidor) se benefician con la utilización de hilos.
- Los hilos proveen un mecanismo que permite a procesos secuenciales hacer llamadas al sistema bloqueantes mientras que también logra paralelismo.

# HILOS: EJEMPLO ARQUITECTURA DE SERVIDOR



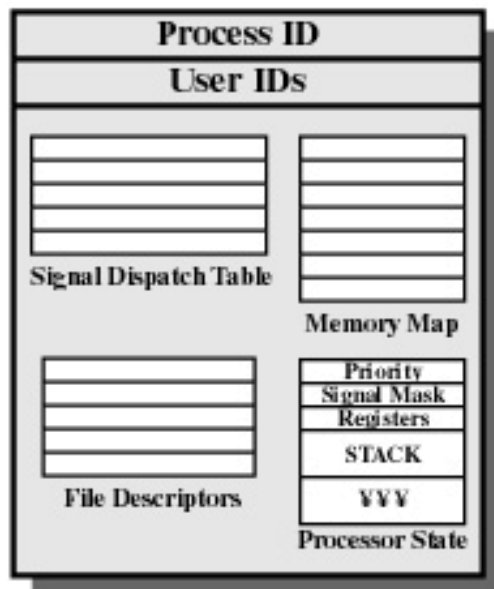


# HILOS: EJEMPLO PROCESADOR DE TEXTO

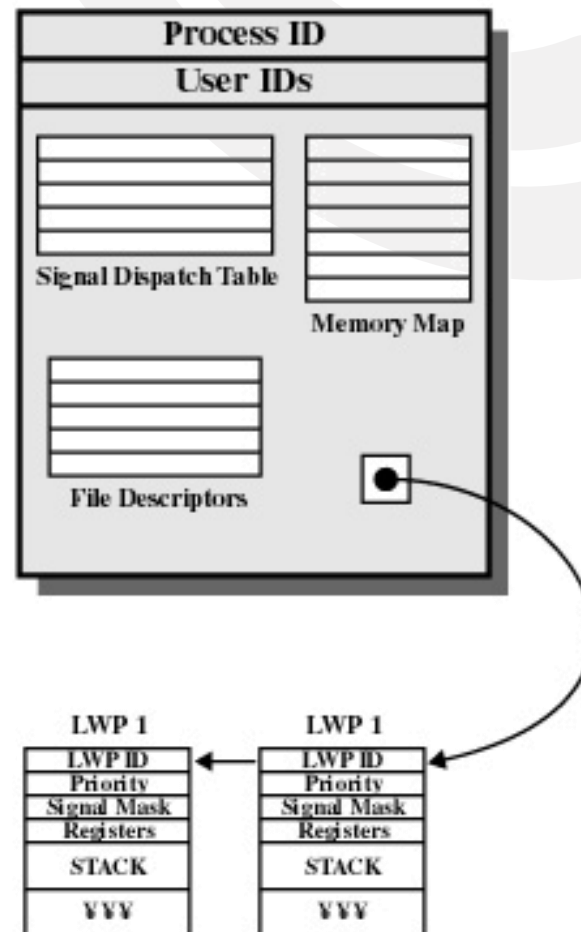


# PROCESOS MONO Y MULTITHILOS

## UNIX Process Structure



## Solaris Process Structure



# BENEFICIOS

- Capacidad de Respuesta
- Compartir Recursos
  - Dado que los hilos dentro de un mismo proceso comparten memoria y archivos, pueden comunicarse unos con otros sin invocar al kernel
- Economía
  - Toma menos tiempo crear un nuevo hilo que un proceso
  - Menos tiempo terminar un hilo que un proceso
  - Menos tiempo en conmutar entre dos hilos dentro del mismo proceso
- Utilización de Arquitecturas Multiprocesador
- Escalabilidad

# PROGRAMACIÓN MULTICORE

Los sistemas multicore ponen presión sobre los programadores, estos desafíos incluyen:

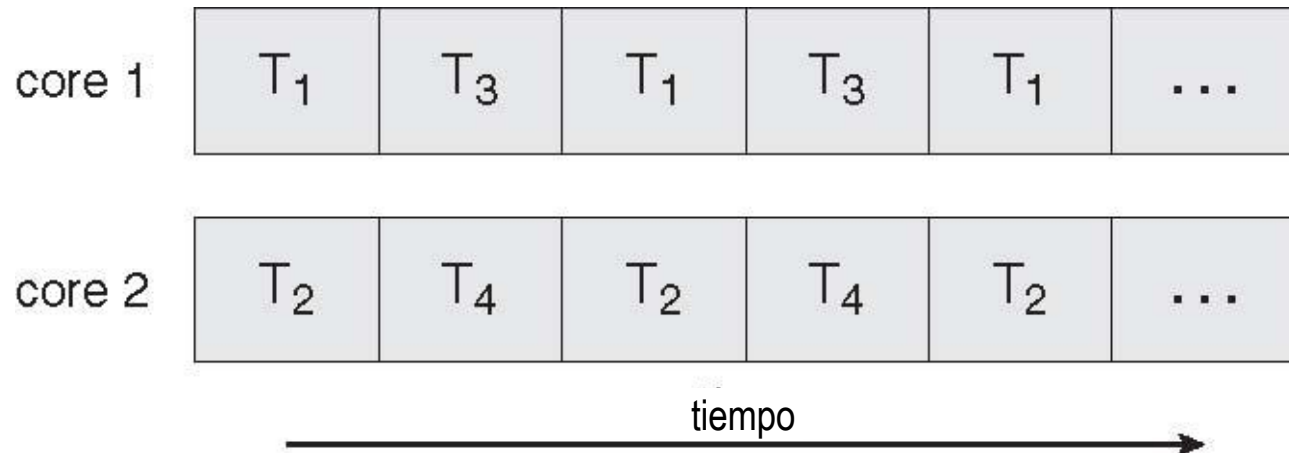
- Dividir actividades
- Balance
- Partición de datos
- Dependencia de los datos
- Verificación y depuración

# EJECUCIÓN CONCURRENT Y PARALELA

## Un solo núcleo

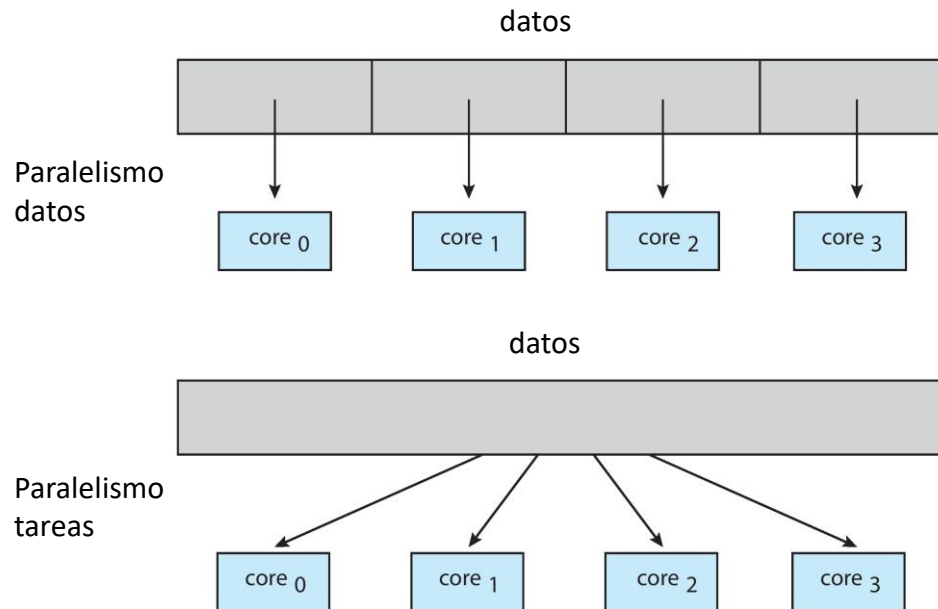


## Múltiples núcleos



# PROGRAMACIÓN MULTICORE

- Tipos de paralelismo
  - **Paralelismo Datos** – distribuye subconjuntos de los mismos datos en varios núcleos, la misma operación en cada uno
  - **Paralelismo Tareas** – distribuir hilos (threads) en los núcleos, cada hilo realiza una operación única



# AGENDA

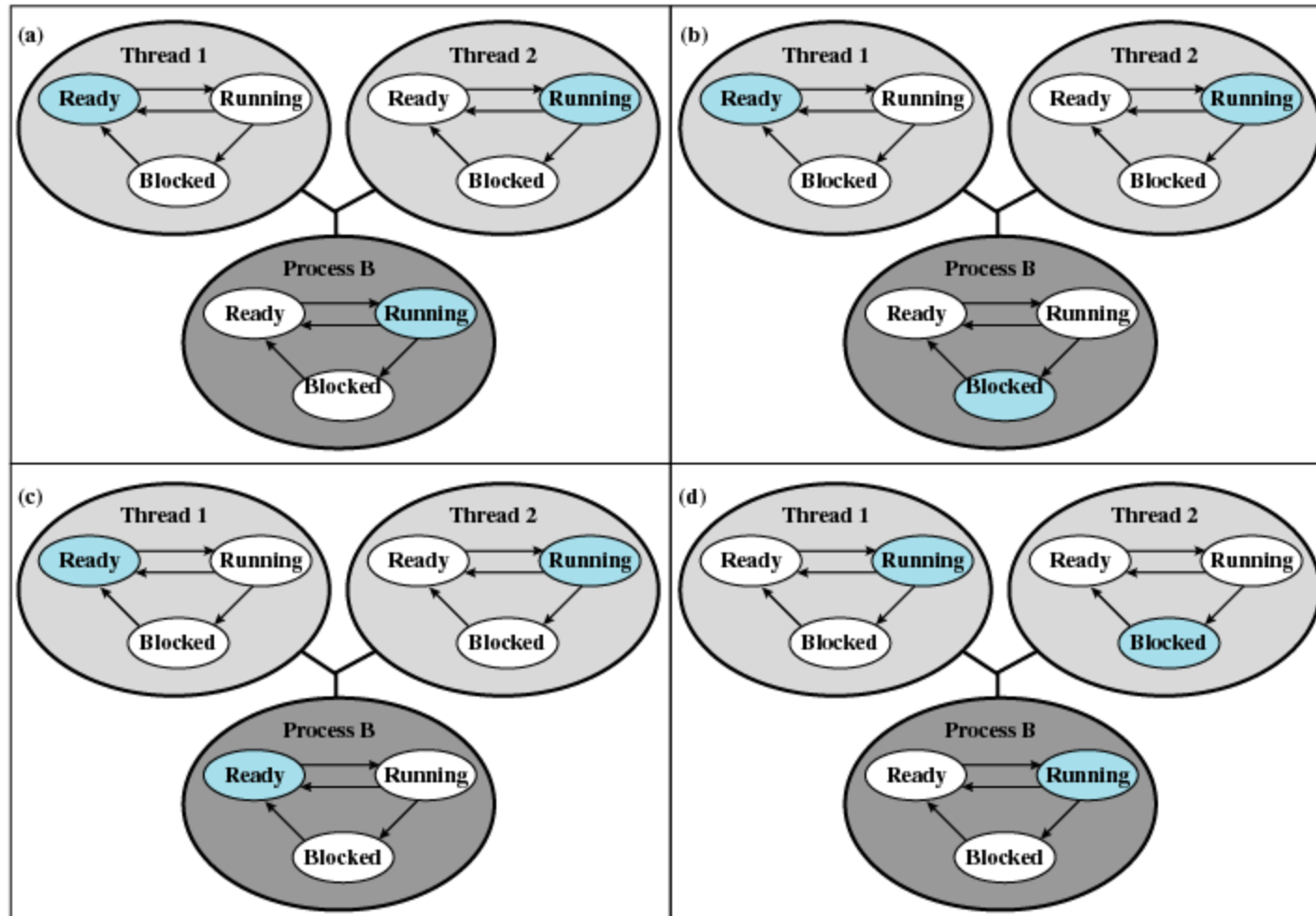
1. Revisión
2. **Clasificación de Hilos**
3. Modelos Multihilados
4. Librerías de Hilos
5. Hilos en Linux

# HILOS - CLASIFICACIÓN

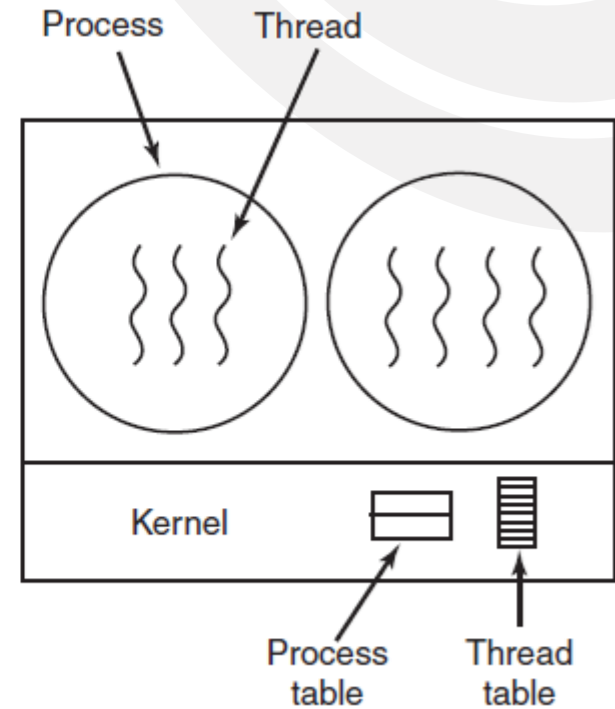
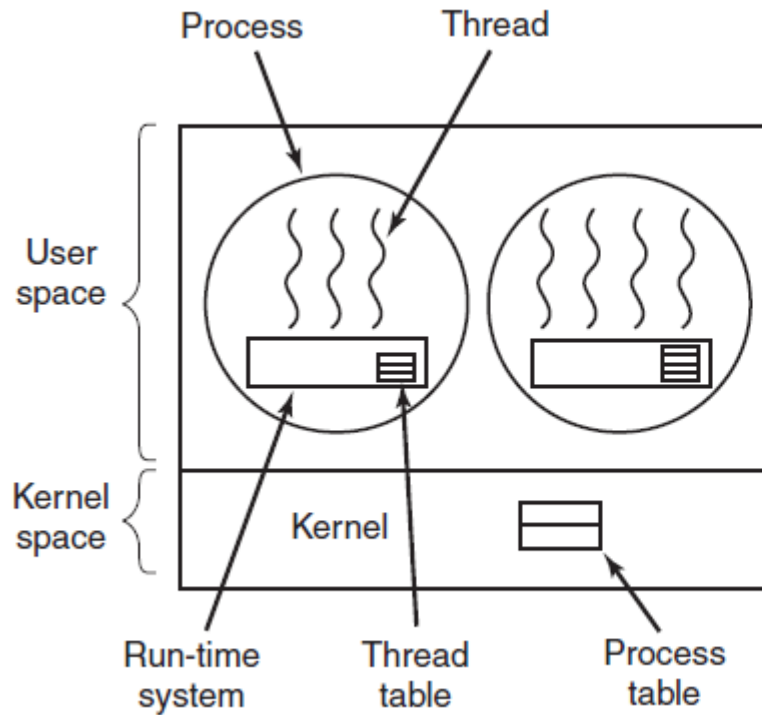
- A nivel de **USUARIO** – la administración es realizada por librerías a nivel de usuario.
  - Tres librerías primarias:
    - POSIX Pthreads
    - Win32 threads
    - Java threads
- A nivel de **KERNEL** – la administración es realizada por el sistema operativo
  - Ejemplos:
    - Windows XP/2000/Vista/7/8
    - Solaris (de Sun, ahora Oracle)
    - Tru64 UNIX (de Digital, luego Compaq, finalmente HP)
    - Mac OS X (Apple)



# Hilos a Nivel de Usuario – Relación estado Proceso e Hilo



# HILOS A NIVEL DE USUARIO



# AGENDA

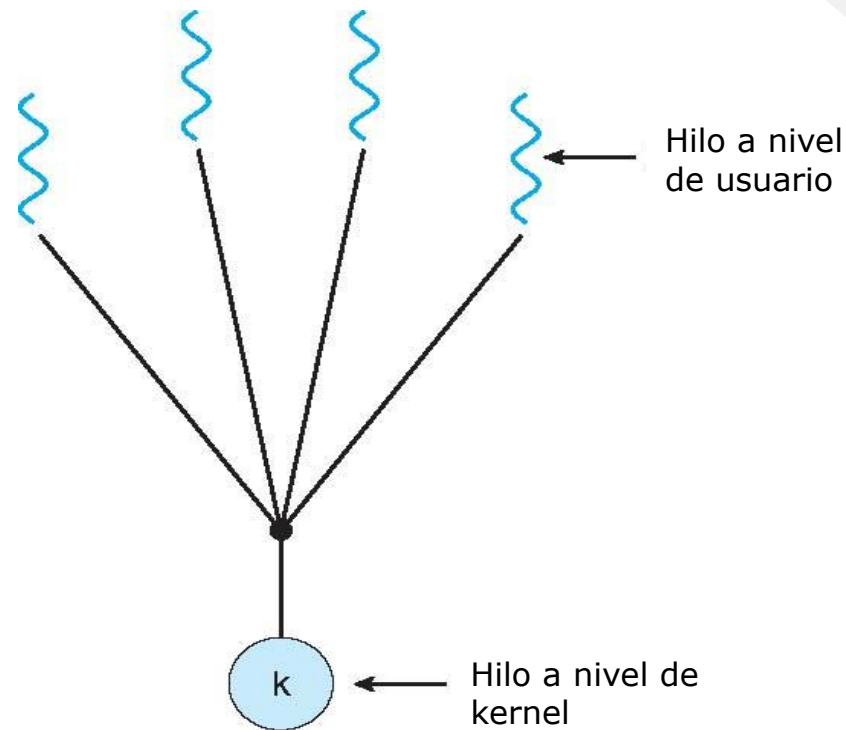
1. Revisión
2. Clasificación de Hilos
3. **Modelos Multihilados**
4. Librerías de Hilos
5. Hilos en Linux

# MODELOS DE MULTIHILADOS

- Muchos-a-Uno
- Uno-a-Uno
- Muchos-a-Muchos

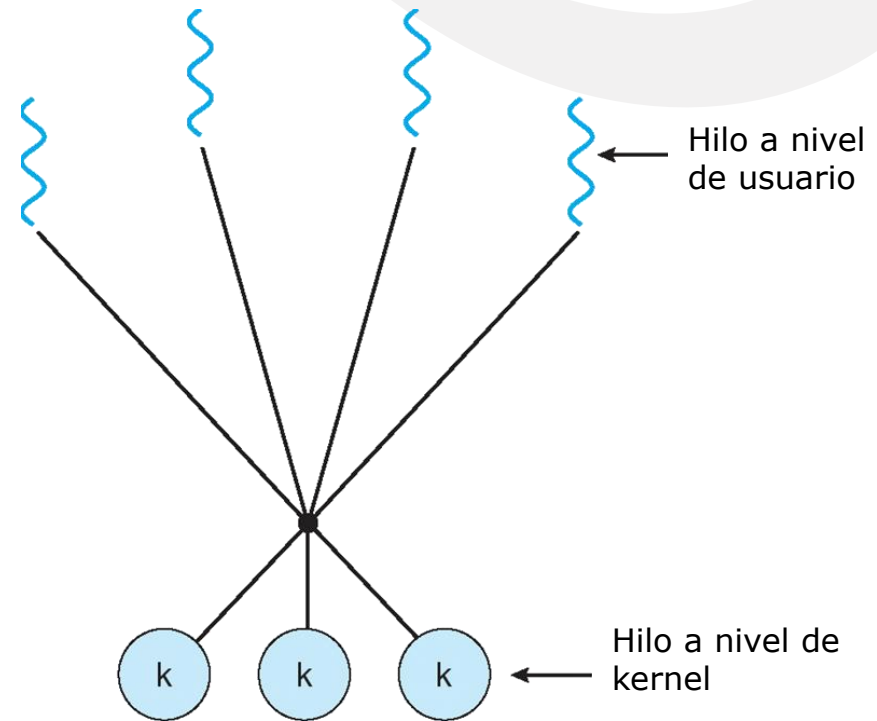
# MUCHOS A UNO

- Muchos hilos a nivel de usuario mapean a un hilo a nivel de kernel.
- Usado en sistemas que no soportan hilos a nivel kernel.



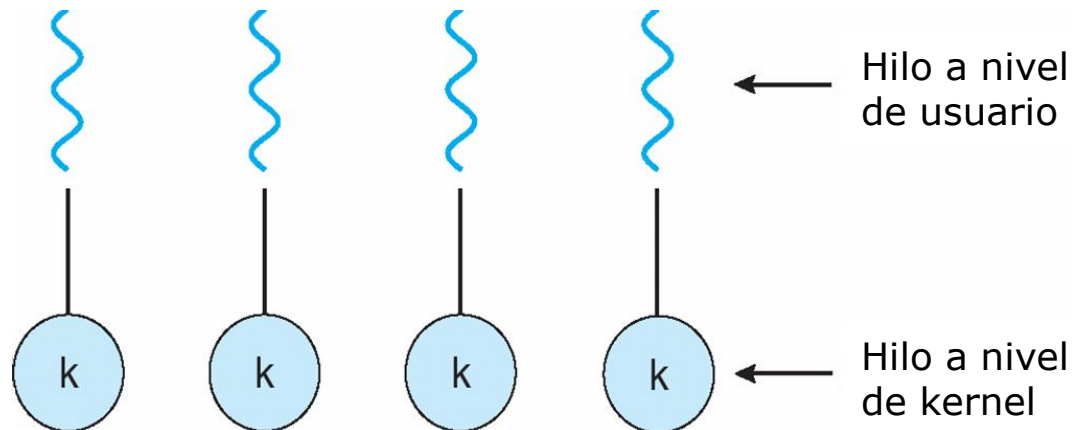
# MUCHOS A MUCHOS

- Permite que muchos hilos a nivel de usuario mapeen a muchos hilos a nivel de kernel
- Permite al SO crear un número suficiente de hilos a nivel de kernel
- Solaris antes de la versión 9
- Windows NT/2000 en adelante con paquete ThreadFiber



# UNO A UNO

- Cada thread nivel usuario mapea a un thread kernel.
- Ejemplos
  - Windows NT/XP/2000 y los que siguen
  - Linux
  - Solaris 9 y los que siguen



# AGENDA

1. Revisión
2. Clasificación de Hilos
3. Modelos Multihilados
- 4. Librerías de Hilos**
5. Hilos en Linux



# LIBRERÍAS DE HILOS

- Las librerías de hilos proveen a los programadores con APIs para crear y administrar hilos
- Dos formas primarias de implementarlas
  - Librerías enteramente en espacio de usuario
  - Librería a nivel de Kernel soportada por el SO

# LIBRERÍA PTHREADS

- Pueden ser provistas sea a nivel de usuario como a nivel de kernel
- Es un standard POSIX (IEEE 1003.1c) API para creación y sincronización de hilos
- Las API especifican el comportamiento de la librería de hilos
- Común en SOs UNIX (Solaris, Linux, Mac OS X)

# MANEJO DE SIGNAL

- Los Signals son usados en UNIX para notificar a un proceso que un particular evento ha ocurrido
- Un **signal handler** es usado para signals a procesos
  1. El Signal es generado por un particular evento
  2. El Signal es enviado a un proceso
  3. El Signal es manejado
- Opciones:
  - Enviar el signal al hilo sobre el cual el signal se aplica
  - Enviar el signal a cada hilo en el proceso
  - Enviar el signal a ciertos hilos en el proceso
  - Asignar un hilo específico para recibir todos los signals al proceso

# CANCELACIÓN DE HILOS

- Terminar un hilo antes que finalice
- Dos propuestas generales:
  - **Cancelación asincrónica** termina el hilo señalado inmediatamente
  - **Cancelación Diferida** permite al hilo señalado verificar periódicamente si debería ser cancelado

# POOLS DE HILOS

Crea un número de hilos en un pool donde esperan por trabajo

Ventajas:

- Usualmente es ligeramente mas rápido servir un requerimiento con un hilo existente que crear uno nuevo
- Permite que el número de hilos de la aplicación sea limitado al tamaño del pool

# AGENDA

1. Revisión
2. Clasificación de Hilos
3. Modelos Multihilados
4. Librerías de Hilos
5. **Hilos en Linux**

# HILOS EN LINUX

- Linux se refiere a ellos como tareas más que como hilos
- La creación de hilos es hecha por la llamada a sistema `clone()`
- `clone()` permite a una tarea hija compartir el espacio de direcciones de la tarea-padre (proceso)

## Bibliografía:

- Silberschatz, A., Gagne G., y Galvin, P.B.; "Operating System Concepts", 7ma Edición 2009; 9na Edición 2012; 10ma Edición 2018.
- Stallings, W. "Operating Systems: Internals and Design Principles", Prentice Hall, 5ta Edición 2005; 6ta Edición 2009; 7ma Edición 2011, 9na Edición 2018.
- Tanenbaum, A.; "Modern Operating Systems", Addison-Wesley, 3ra Edición 2008, 4ta. Edición 2014.