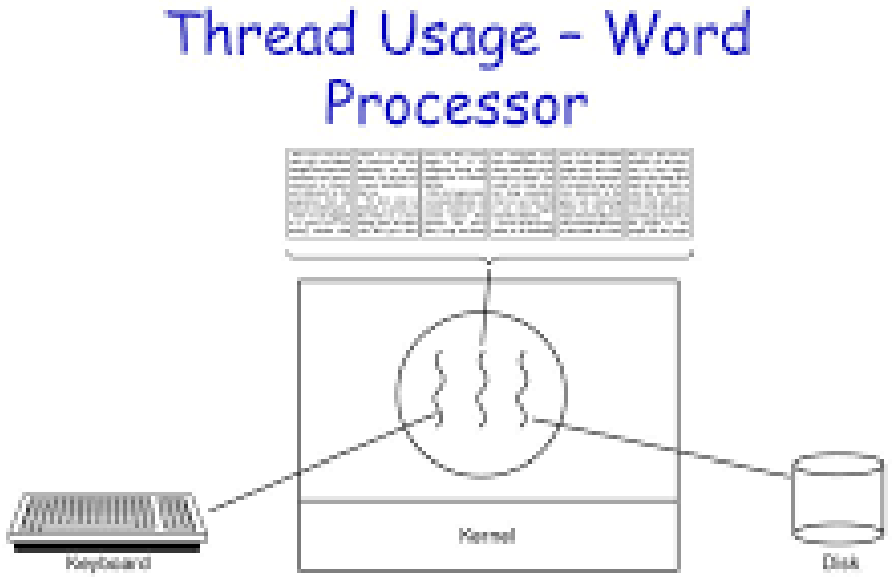
Thread(s)

- A ***Thread*** is the smallest sequence of instructions that can be handled as an independent control program by the OS low level scheduler. **States** as the processes.
- Advantages:
  - Parallelize calculations within the same address space
  - Easier to create/complete
  - Exploit any surplus of computing and I/O resources by overlapping.



A word processor program with three threads.

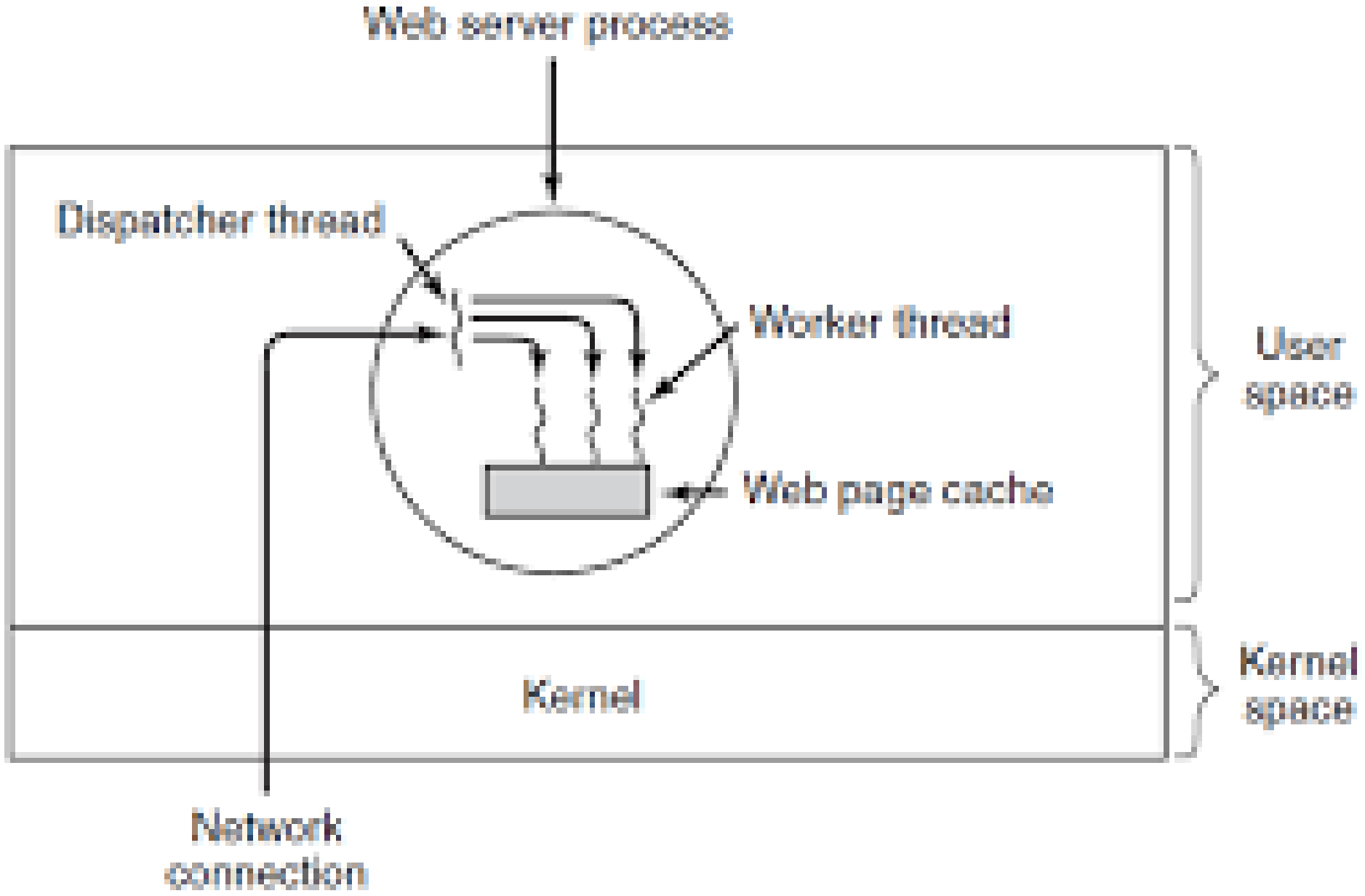


Figure 2-8. A multithreaded Web server.

Models to construct a Server

Model	Characteristics/Features
• Threads	Parallelism, blocking system calls
• Single-Thread	No parallelism , blocking system calls
• Finite-State Machine	Parallelism, non blocking system calls, interrupts

- A process can have: an input thread, a processing thread, and an output thread
- The input thread reads data into an input buffer.
- The processing thread takes data out of the input buffer, processes them, and puts the results in an output buffer.
- The output buffer writes these results back to disk.
- Result: input, output, and processing can all be going on at the same time.
- This model works only if a system call blocks ***only the calling thread***, not the entire process

Processes and Threads
Multithreading: multiple threads in a single process

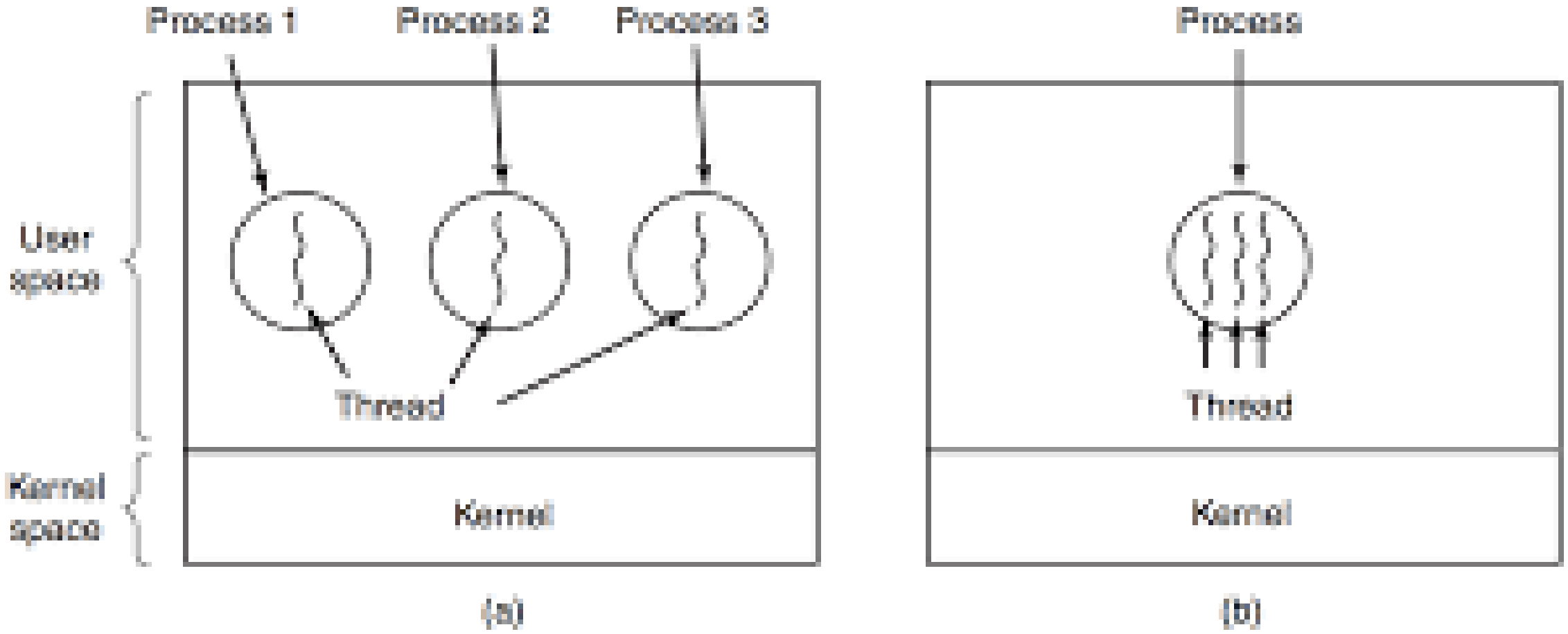


Figure 2-11. (a) Three processes each with one thread. (b) One process with three threads.

Process Items

- Address space
- Global Variables
- Open Files
- Child processes
- Signals & Signal Handlers
- Accounting Information

Thread Items

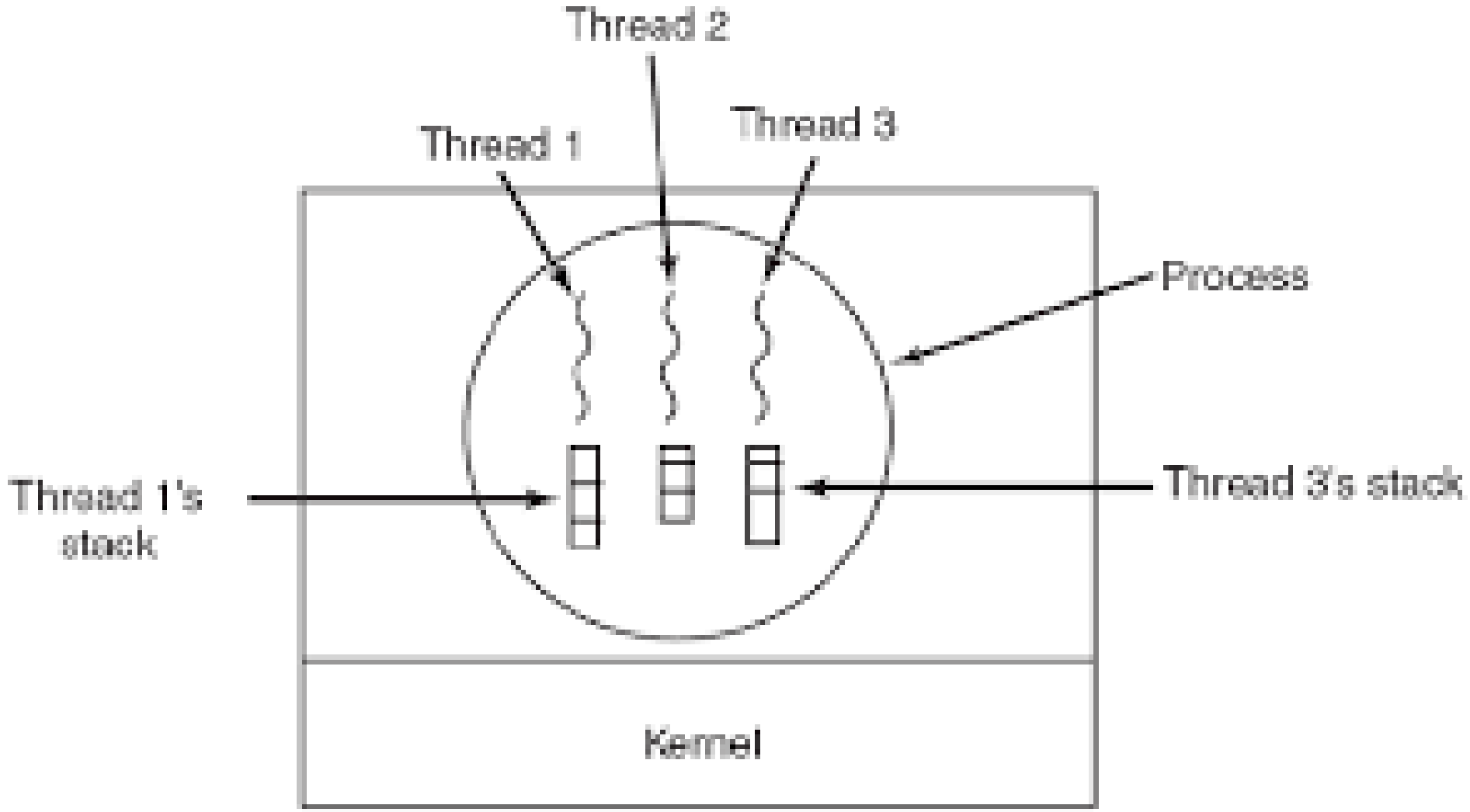
Program Counter

Registers

Stack

State

Threads and their Stacks



IEEE standard 1003.1c (Posix Threads) The threads package is called **Pthreads**

Pthreads function calls (most common)

Process Items	**Thread Items**
• Pthread_create	Create a new thread
• Pthread_exit	Ter minate the calling thread
• Pthread_join	Wait for a specific thread to exit
• Pthread_yield	Release the CPU to let another thread run
• Pthread_attr_init	Create and initialize a thread's attribute structure
• Pthread_attr_destroy	Remove a thread's attribute structure

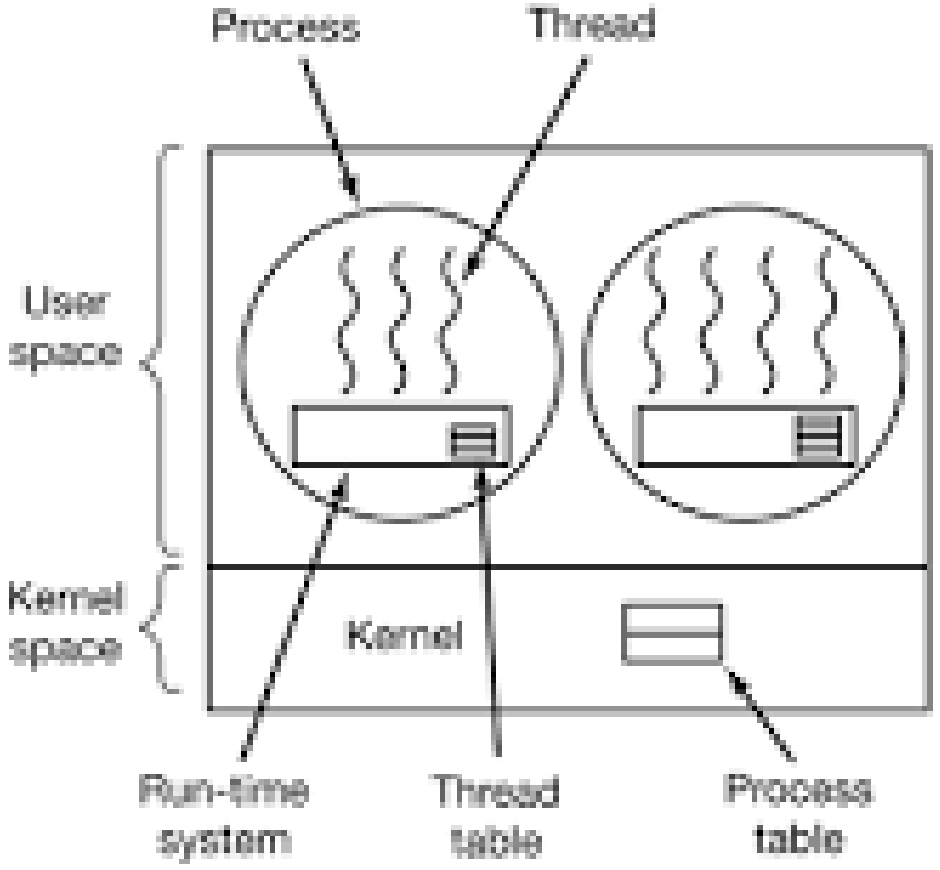
Threads in *User* Space

- The kernel manages the processes (as single-threaded)
- OS without Threads -> Threads by a Library
- Threads are call by a run-time system procedure
- Faster tread-switching (no trap, no context switching)
- Process: ***custom*** tread mgt algrorithm

- Disadvantages
  - Blocking system calls, e.g. reading from the keyboard (if nothing, then it blocks)
    - Select: check if it is safe
    - Read: following a safe select.

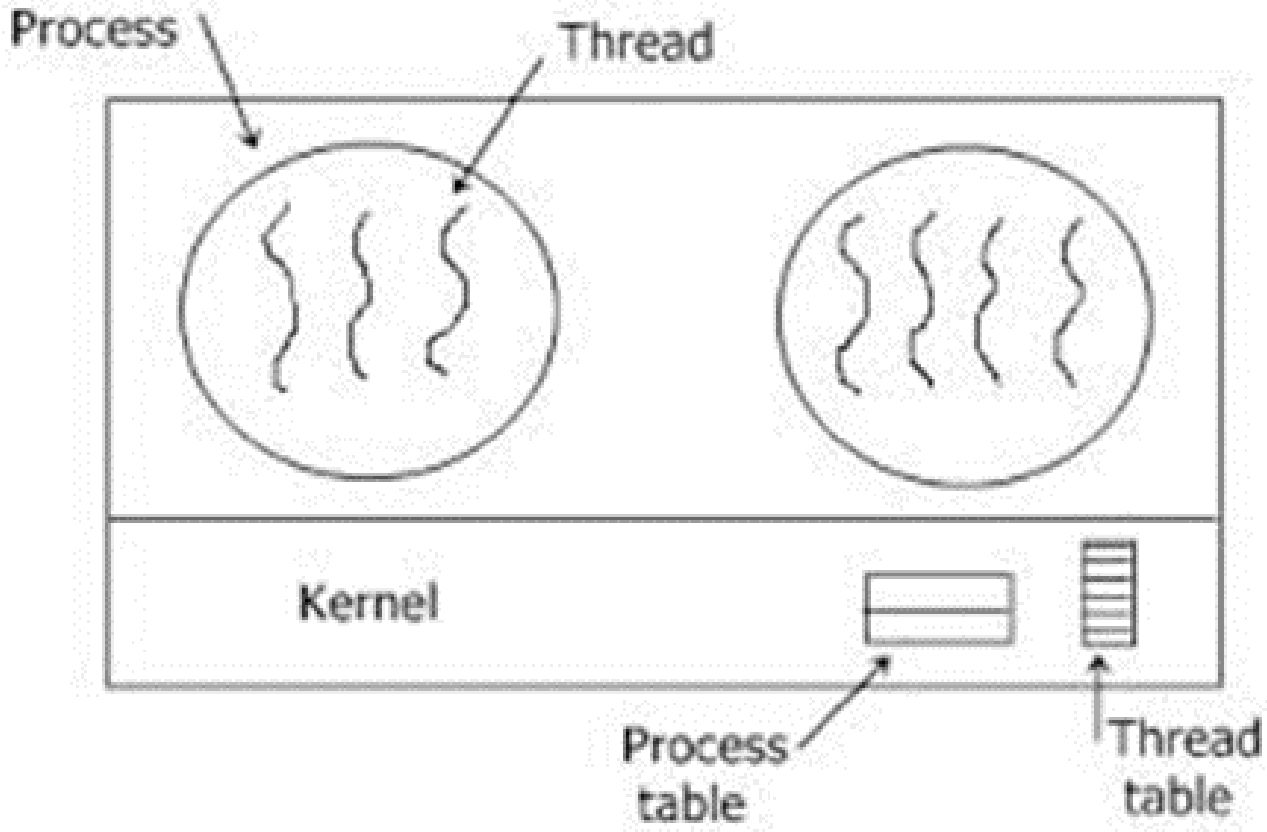
    Additional code:

    Jacket or ***Wrapper***



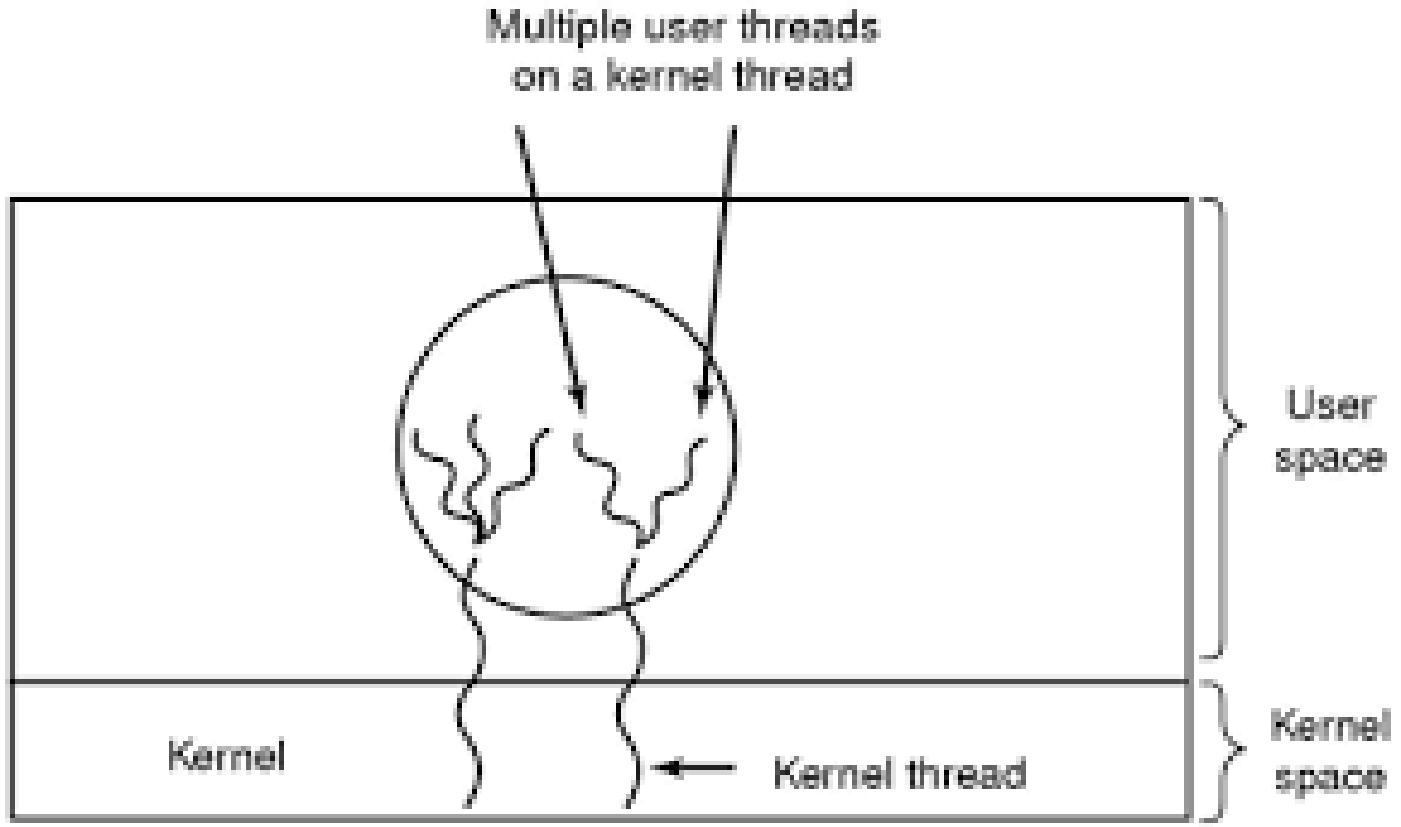
Threads in the Kernel Space

- No ***run-time system*** needed
- New threads: by kernel call
- Threads are implemented as system calls (costly)
- Threads switch: different processes
- Threads destroy: *not runnable* to keep
  their data structure for another
  of the same type
- Upon blocked system call another of the
  same process will run
- Signals directed to processes:
  associate each thread with signals



Hybrid Implementations

- The programmer optimizes the combination
- Flexibility



Pop-up Threads

- Distributed Systems
- Creation of a new thread when message arrives
  - Before message arrives
  - After message arrives
- Most cases in *Kernel space*