Threads

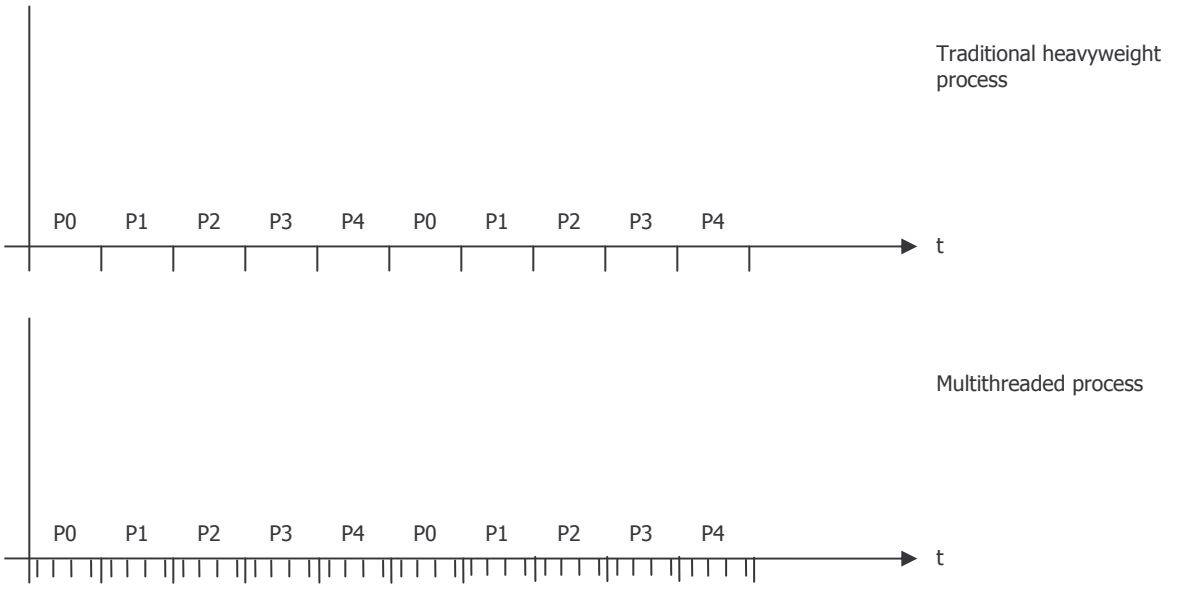
Thread
A path of control in an application
Also, a unit of CPU utilization that allows us to compare architectures

Heavyweight Process
A traditional process with a single thread.

Ex: main() executing

Lightweight Process
A thread in a multithreaded process

It is lightweight because it does not carry around the baggage of the code, data, and files. It only contains the registers and the stack.



Windows, Linux, and Mac OS X 10 are all multithreaded operating systems. Their kernels, themselves, are threaded.

Benefits of using Threads
1. Responsiveness
2. Resource sharing
    a. Gets rid of overhead that is present when forking a child – everything gets copied.
    b. Sharing code, data, and files allows better use of resources
3. Economy of time
4. Multiprocessor architecture
    a. A set of threads can be dispatched to each processor

Supporting Second Level of Context Switching

1. **User Threads**
    a. Because there is no kernel support, the process, itself, must implement a scheduler to switch threads on and off.
    b. This can use another interrupt or a large loop which calls several different subroutines.
2. **Kernel Threads**
    a. Offers complete kernel support. The kernel configures the second context switch using either a second interrupt or a faster interrupt.
    b. Implemented using threading libraries provided by the OS. These libraries allow you to write the application. One of the most common on Unix/Linux is the `pthreads` library (POSIX).
    c. This is the way of doing this, because all modern operating systems support threads

Most modern OSes have a limit to the number of threads. The exact limit varies based on the OS.

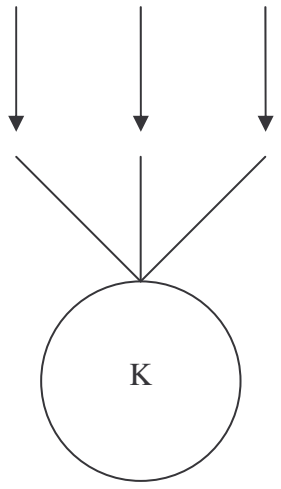
Java's interpreter can act as a kernel supporting threads. It provides an API for threading.

Strategies for Implementing Threads

There must be a process of creating/getting threads and associating them with kernel activity.

Many-to-one

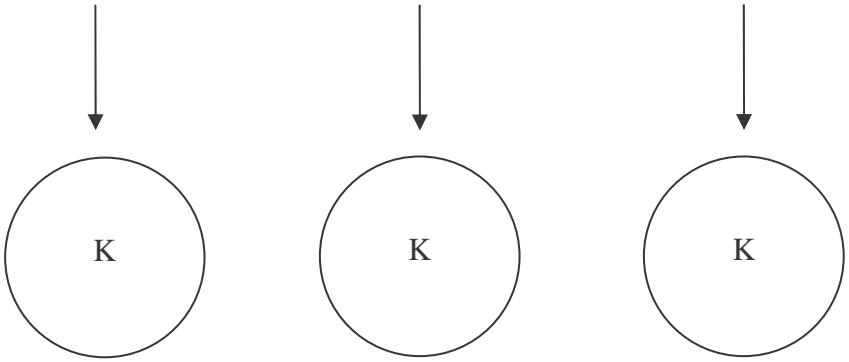
When your process is loaded into memory



Disadvantage – all blocking (waiting for I/O) stops the threads

User threads controlled by a single kernel thread

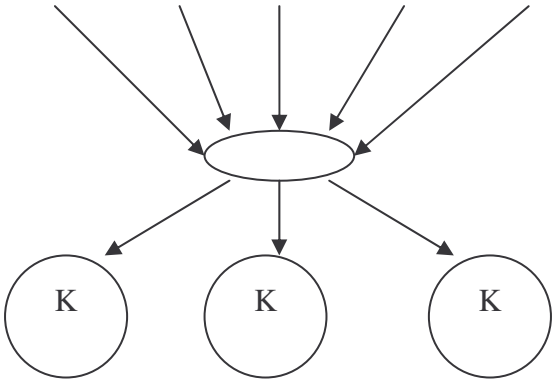
One-to-one



Advantage: Blocking does not hurt anything.
Disadvantage – each kernel thread takes up additional resources

Each user thread controlled by a single kernel thread

Many-to-many



We are asking the kernel to support our context switch