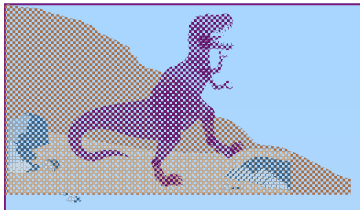
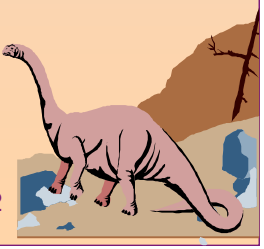
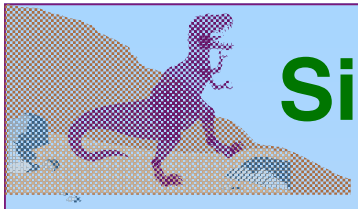


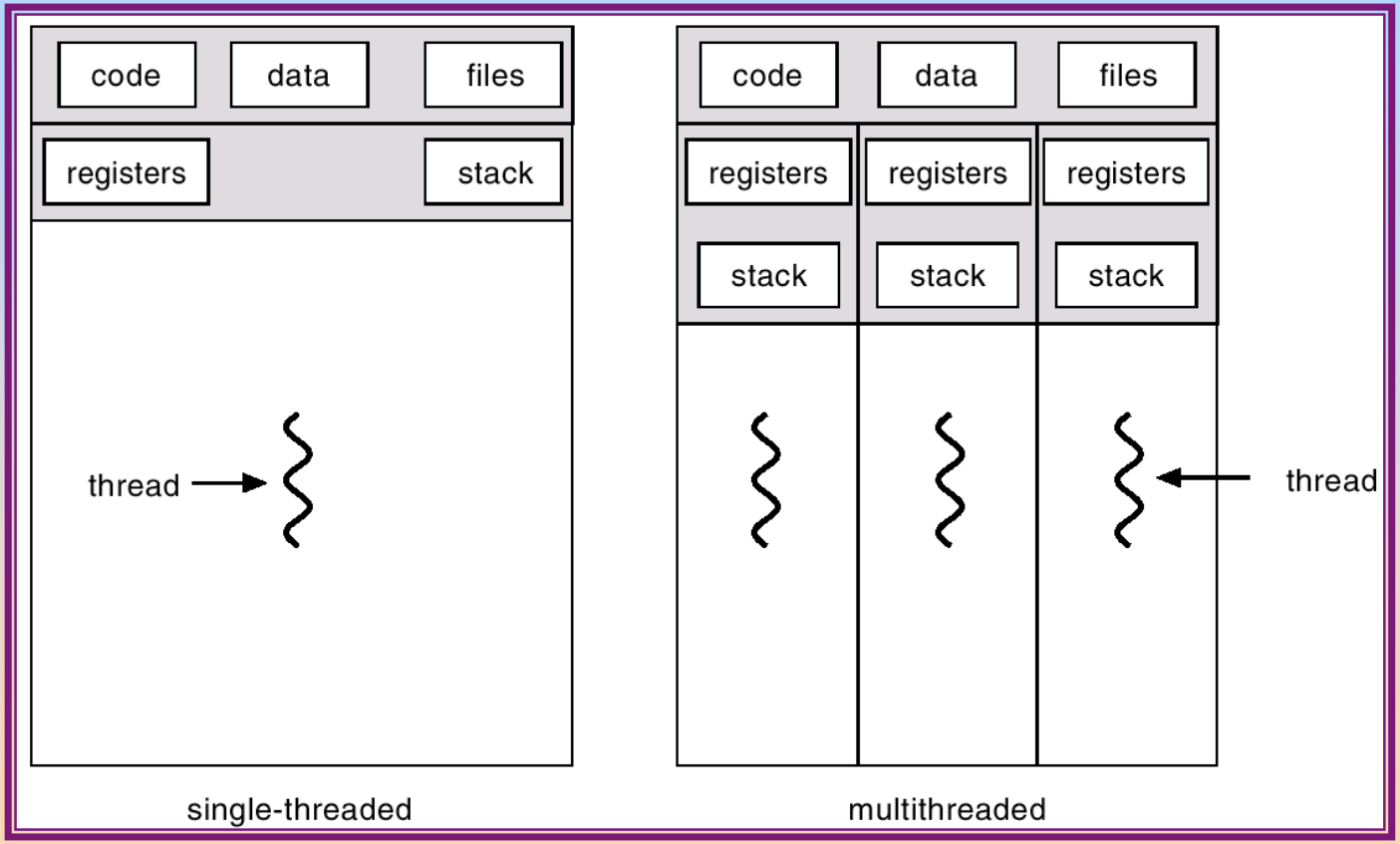
Chapter 5: Threads

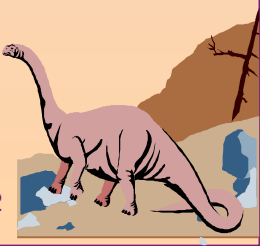
- Overview
- Multithreading Models
- Threading Issues
- Pthreads
- Solaris 2 Threads
- Windows 2000 Threads
- Linux Threads
- Java Threads

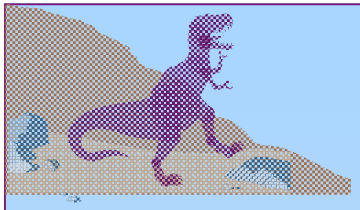




Single and Multithreaded Processes

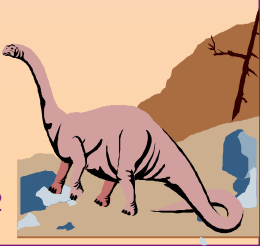


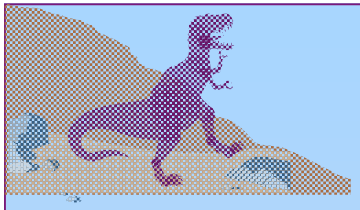




Benefits

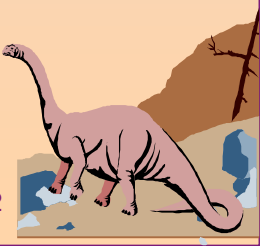
- Responsiveness
- Resource Sharing
- Economy
- Utilization of MP Architectures

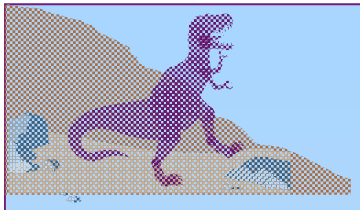




User Threads

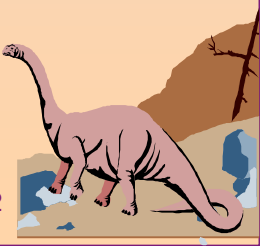
- Thread management done by user-level threads library
- Examples
  - POSIX *Pthreads*
  - Mach *C-threads*
  - Solaris *threads*

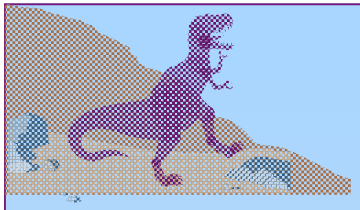




Kernel Threads

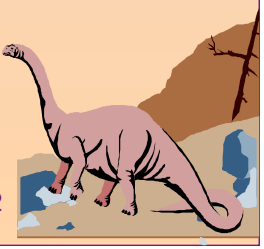
- Supported by the Kernel
- Examples
  - Windows 95/98/NT/2000
  - Solaris
  - Tru64 UNIX
  - BeOS
  - Linux

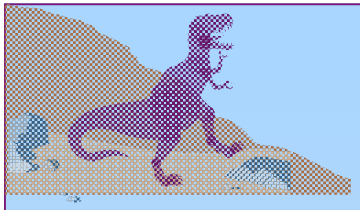




Multithreading Models

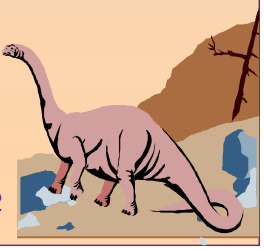
- Many-to-One
- One-to-One
- Many-to-Many

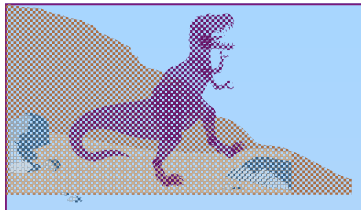




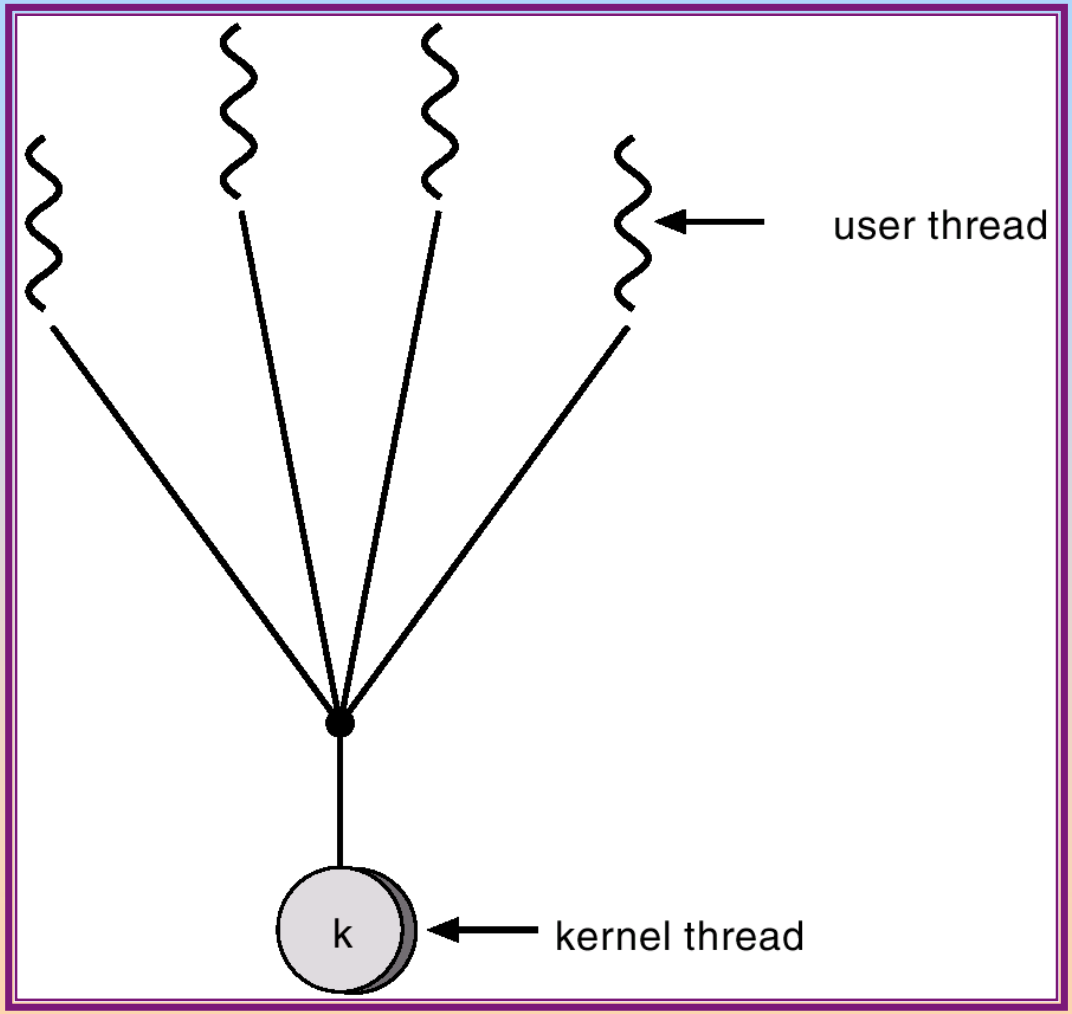
Many-to-One

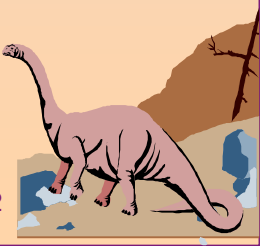
- Many user-level threads mapped to single kernel thread.
- Used on systems that do not support kernel threads.

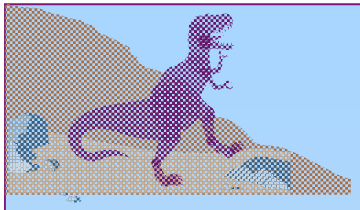




Many-to-One Model

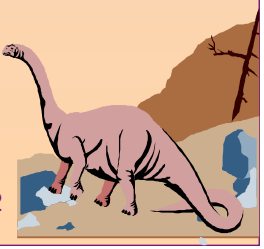




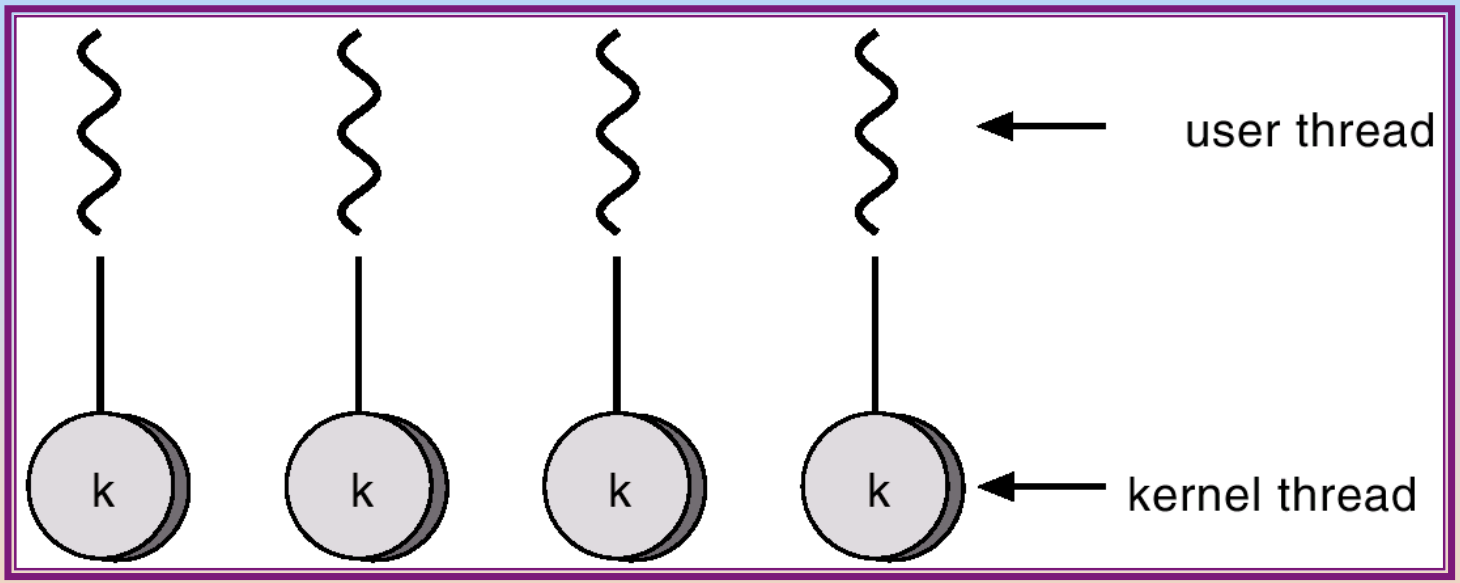


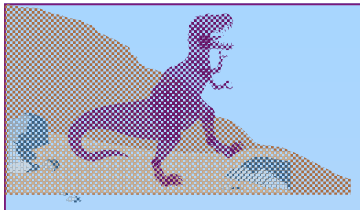
One-to-One

- Each user-level thread maps to kernel thread.
- Examples
  - Windows 95/98/NT/2000
  - OS/2



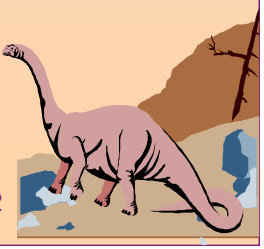
One-to-one Model

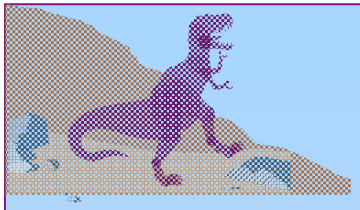




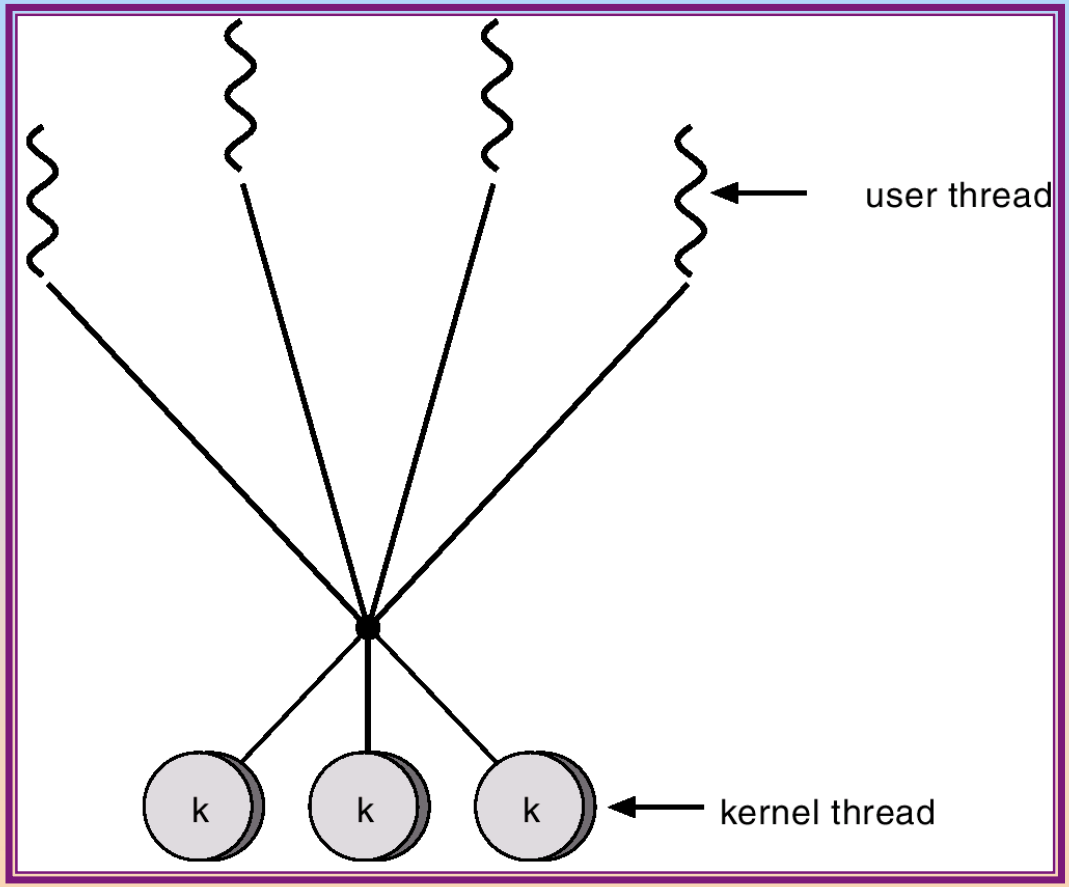
Many-to-Many Model

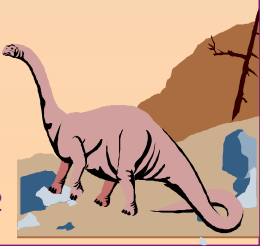
- Allows many user level threads to be mapped to many kernel threads.
- Allows the operating system to create a sufficient number of kernel threads.
- Solaris 2
- Windows NT/2000 with the *ThreadFiber* package

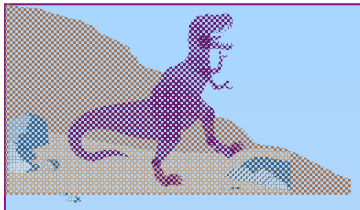




Many-to-Many Model

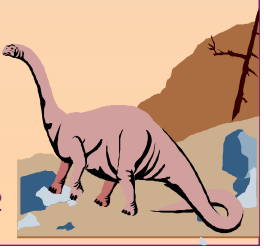


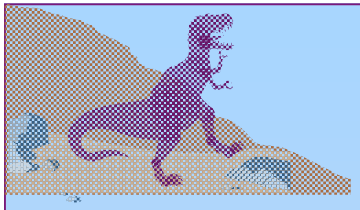




Threading Issues

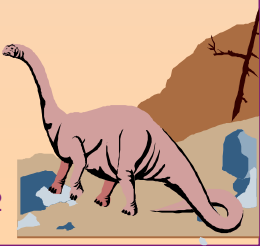
- Semantics of fork() and exec() system calls.
- Thread cancellation.
- Signal handling
- Thread pools
- Thread specific data

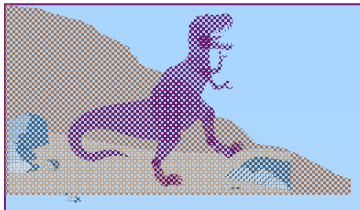




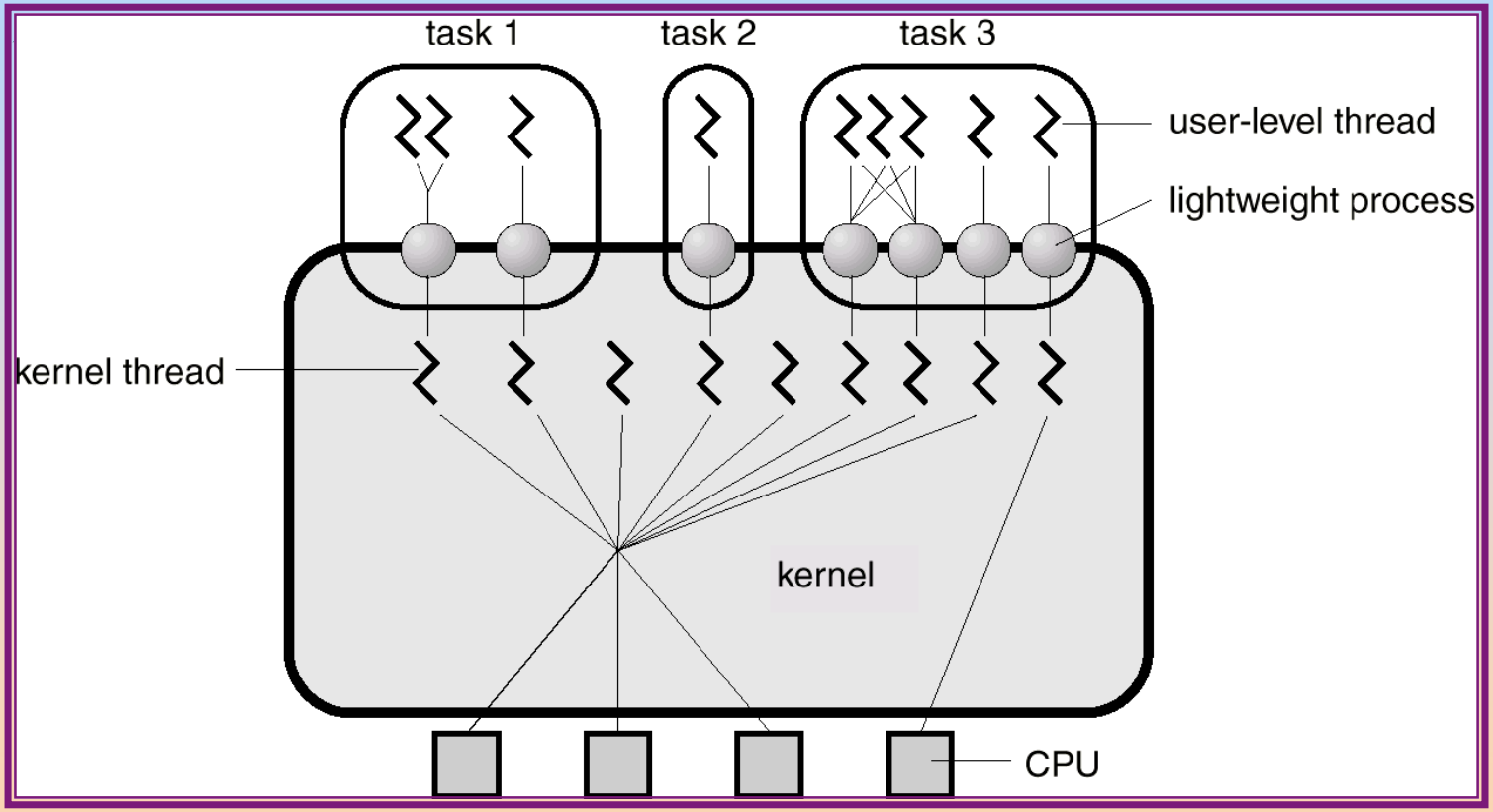
Pthreads

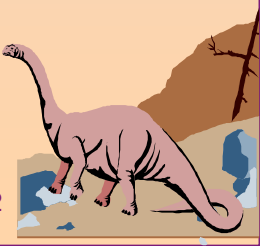
- a POSIX standard (IEEE 1003.1c) API for thread creation and synchronization.
- API specifies behavior of the thread library, implementation is up to development of the library.
- Common in UNIX operating systems.

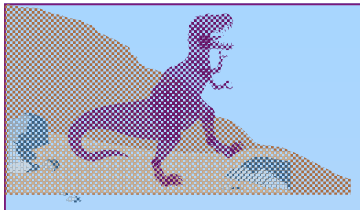




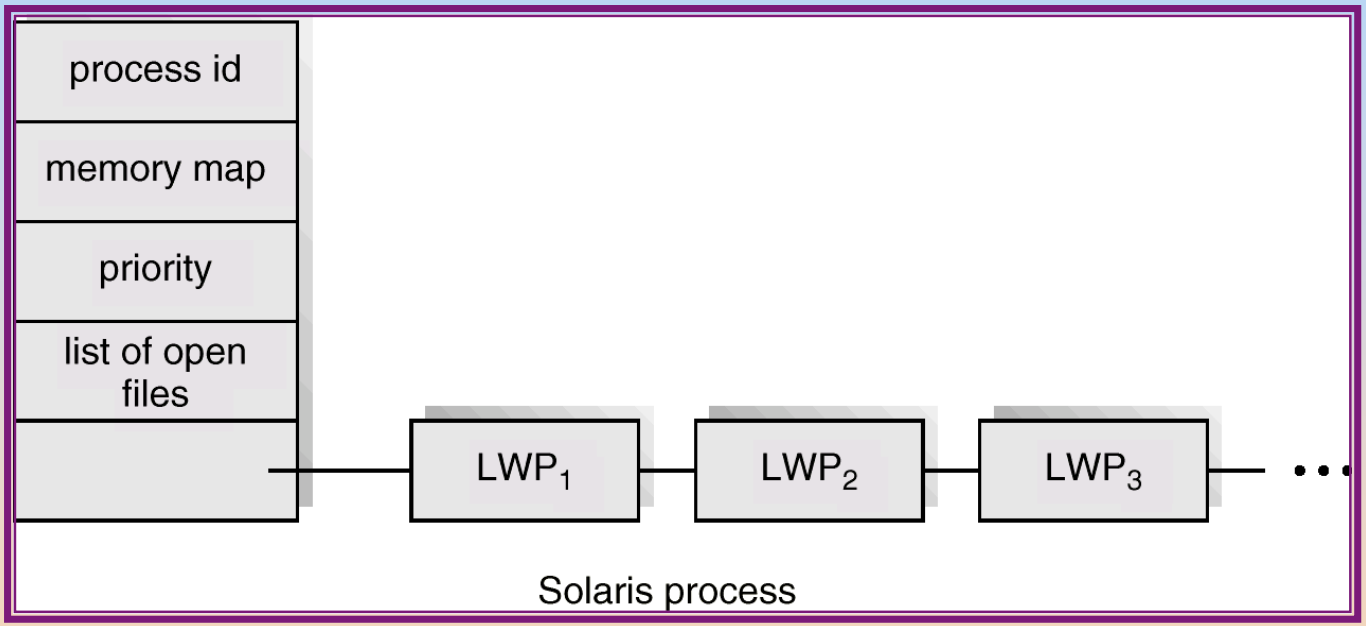
Solaris 2 Threads

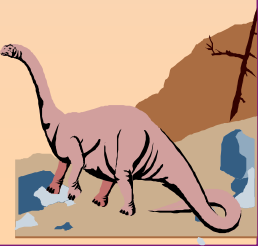


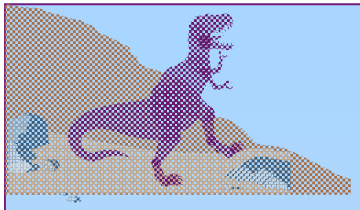




Solaris Process

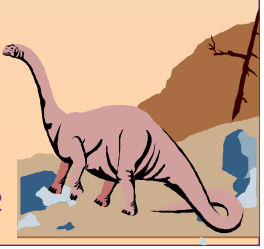


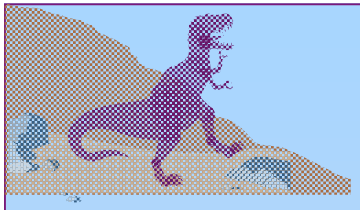




Windows 2000 Threads

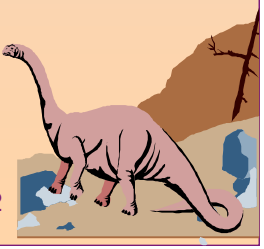
- Implements the one-to-one mapping.
- Each thread contains
  - a thread id
  - register set
  - separate user and kernel stacks
  - private data storage area

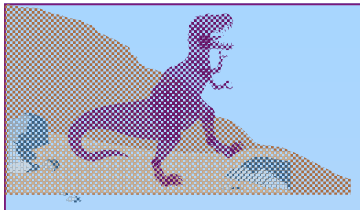




Linux Threads

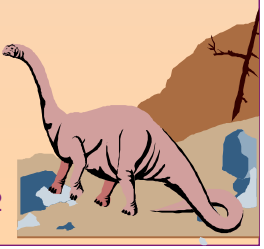
- Linux refers to them as *tasks* rather than *threads*.
- Thread creation is done through clone() system call.
- Clone() allows a child task to share the address space of the parent task (process)

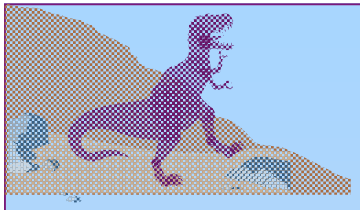




Java Threads

- Java threads may be created by:
  - Extending Thread class
  - Implementing the Runnable interface
- Java threads are managed by the JVM.





Java Thread States