

Chapter 5: Threads

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

Operating System Concepts 5.1 Silberschatz, Galvin and Gagne ©2002

Single and Multithreaded Processes

Operating System Concepts 5.2 Silberschatz, Galvin and Gagne ©2002

Benefits

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

Operating System Concepts 5.3 Silberschatz, Galvin and Gagne ©2002

User Threads

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

Operating System Concepts 5.4 Silberschatz, Galvin and Gagne ©2002

Kernel Threads

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

Operating System Concepts 5.5 Silberschatz, Galvin and Gagne ©2002

Multithreading Models

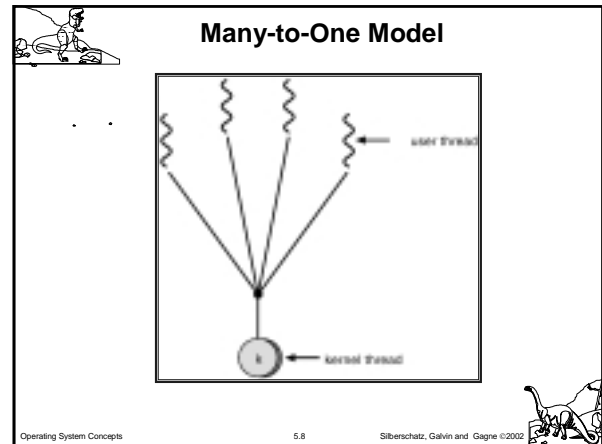
- Many-to-One
 - ⊗ "Green Threads"
 - ⊗ Easiest for OS to manage
- One-to-One
 - ⊗ Windows NT/2000
 - ⊗ Limits number of threads created
- Many-to-Many
 - ⊗ Solaris (Sun/Unix)
 - ⊗ Balances two approaches to allow unlimited threads

Operating System Concepts 5.6 Silberschatz, Galvin and Gagne ©2002

Many-to-One

- Many user-level threads mapped to single kernel thread.
 - ☞ (i.e. one kernel process supports multiple user threads)
- Used on systems that do not support kernel threads.

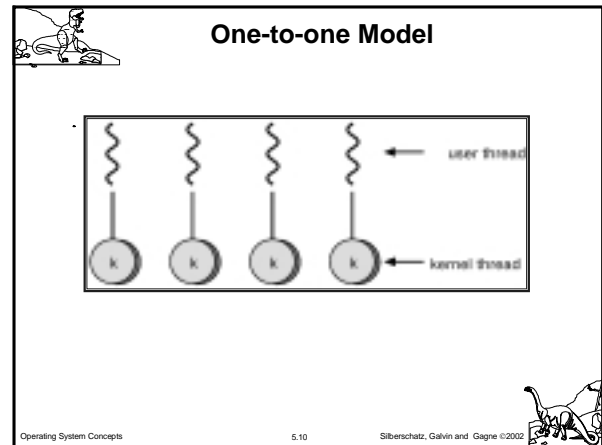
Operating System Concepts 5.7 Silberschatz, Galvin and Gagne ©2002



One-to-One

- Each user-level thread maps to its own kernel thread.
- Limits must be placed on the total number of threads allowed for user processes
- 1 to 1 provides the fastest execution of threads but may waste thread resources (i.e. blocked threads)
- Examples
 - Windows 95/98/NT/2000
 - OS/2

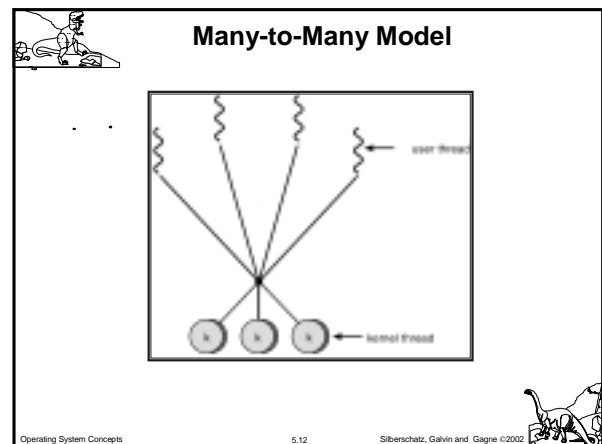
Operating System Concepts 5.9 Silberschatz, Galvin and Gagne ©2002



Many-to-Many Model

- Allows many user level threads to be mapped to many kernel threads.
- Made possible by the use of a *Thread Pool*
- Allows the operating system to create a sufficient number of kernel threads.
 - ☞ Only practical limit is amount of available memory
- Solaris 2
- Windows NT/2000 with the *ThreadFiber* package

Operating System Concepts 5.11 Silberschatz, Galvin and Gagne ©2002



Threading Issues

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

Operating System Concepts 5.13 Silberschatz, Galvin and Gagne ©2002

Pthreads

- à 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 / Linux operating systems.

Operating System Concepts 5.14 Silberschatz, Galvin and Gagne ©2002

Solaris 2 Threads

The diagram illustrates the Solaris 2 threads architecture. At the top, three 'TASK' boxes (TASK 1, TASK 2, TASK 3) contain 'user-level thread' icons. These threads are grouped into 'lightweight process' containers. Below these, 'kernel thread' icons are shown, which are connected to 'CPU' blocks at the bottom. A central 'kernel' box is also shown with lines connecting to the kernel threads.

Operating System Concepts 5.15 Silberschatz, Galvin and Gagne ©2002

Solaris Process

The diagram shows the structure of a Solaris process. It consists of a box containing: 'process id', 'memory map', 'priority', and 'list of open files'. Below this box, a sequence of 'LWP₁', 'LWP₂', 'LWP₃', and an ellipsis '...' are shown, representing the Lightweight Processes within the process.

Operating System Concepts 5.16 Silberschatz, Galvin and Gagne ©2002

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

Operating System Concepts 5.17 Silberschatz, Galvin and Gagne ©2002

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)

Operating System Concepts 5.18 Silberschatz, Galvin and Gagne ©2002

Java Threads

- Java threads may be created by:
 - ⊕ Extending Thread class
 - ⊕ Implementing the Runnable interface
- Java threads are managed by the JVM.
 - ⊕ A fully object-oriented treatment of threads

Operating System Concepts 5.19 Silberschatz, Galvin and Gagne ©2002

