Chapter 5: Threads

- 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 NT/2000/XP
  - Solaris 2
  - Mach
  - Linux 2.2

Multithreading Models

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

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

One-to-One

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

Many-to-Many Model

- Allows many user-level threads to map to many kernel threads.
- Allows the operating system to create a sufficient number of kernel threads.
- Examples:
  - Solaris 2
  - Windows NT/2000/XP with the ThreadFiber package.
Many-to-Many Model

Threading Issues

- Semantics of fork() and exec() system calls
  - Does fork duplicate all threads of process?
  - Does exec replace thread or process?
- Thread cancellation
  - Common in multi-threaded applications
  - Might leave data inconsistent
- Signal handling
  - Which thread gets the signal?
- Thread pools
  - Pre-made threads to speed thread creation
- Thread specific data
  - Shouldn’t be, but is useful

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

- Task 1, Task 2, Task 3
- Kernel thread
- User-level thread
- Light-weight process
- CPU

Solaris Process

- Process id
- Memory map
- Priority
- List of open files
- LWP1, LWP2, LWP3

Windows XP 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