The process model introduced in Chapter 3 assumed that a process was an executing program with a single thread of control. Virtually all modern operating systems, however, provide features enabling a process to contain multiple threads of control. In this chapter, we introduce many concepts associated with multithreaded computer systems, including a discussion of the APIs for the Pthreads, Windows, and Java thread libraries. We look at a number of issues related to multithreaded programming and its effect on the design of operating systems. Finally, we explore how the Windows and Linux operating systems support threads at the kernel level.

**Bibliographical Notes**

Threads have had a long evolution, starting as “cheap concurrency” in programming languages and moving to “lightweight processes,” with early examples that included the Thoth system ([Cheriton et al. (1979)]) and the Pilot system ([Redell et al. (1980)]). [Binding (1985)] described moving threads into the UNIX kernel. Mach ([Accetta et al. (1986)], [Tevanian et al. (1987)]), and V ([Cheriton (1988)]) made extensive use of threads, and eventually almost all major operating systems implemented them in some form or another.


An analysis of an optimal thread-pool size can be found in [Ling et al. (2000)]. Scheduler activations were first presented in [Anderson et al. (1991)], and [Williams (2002)] discusses scheduler activations in the NetBSD system.
Chapter 4  Threads


Bibliography


