Book Reviews

Thuan Q. Pham, Pankaj K. Garg

Multithreaded Programming with Win32

Prentice Hall PTR, 1999, pp. IX, 219, ISBN 0–13–010912–6

Until recently, concurrent programming required, instead of multiple threads, multiple processes, each with a single thread of execution, running concurrently on a multitasking operating system. Multithreading is a way to design and implement parallel application programs. The major difference between a thread and a process is in common address space. Threads have the same address space which makes communication and data sharing among threads efficient. The goal of multithreaded programming concept is to achieve a greater speed from multiprocessor computers.

The purpose of this book is to introduce the concept of multithreaded programming and to demonstrate how multithreaded programs work on several dozens of code examples. CD-ROM is included. It contains the source code and executable programs covered in the book.

Chapter 1 introduces the basic concepts, history and advantages of multithreaded programming. It also presents an example of a multithreaded program.

The second chapter discusses the concepts of processes and threads, illustrates thread management functions and examines several synchronization objects for multithreaded programming.

The basic synchronization techniques are illustrated in Chapter 3. The classic concurrency problems of producer-consumer, bounded-buffer and readers- writers are demonstrated.

Following two chapters introduce monitors. Monitors are used to simplify multithreaded programs by encapsulating shared data, by centralizing low-level details of mutual exclusion and synchronization. Chapter 4 describes the structure and properties of monitors and benefits of monitors. Chapter 5 shows how to implement monitors using C++ classes synchronization primitives.

Chapter 6 covers the problem of system deadlocks. Theoretically are investigated resource requirements and conditions that cause deadlock, and then, using the dining philosophers problem as an example, the commonly used strategies and techniques to detect, prevent, and avoid system deadlock are discussed and demonstrated.

Multithreading implementation architecture in relation to the operating system, including user-level threads, kernel-level threads, multiplexed threads and kernel-supported user-level threads are presented in Chapter 7. Advantages, limitations and implications in application program design are discussed for each implementation.

Chapter 8 discusses the implementation problem for an application with multiple threads. Several illustrative programming models were presented which include work groups model, manager-worker model, deferred computation, pipelines and WorkCrew. Every model is demonstrated with a sample program.

Usage of multiple threads in distributed computing is examined in Chapter 9. The client-server model is briefly described and the basic steps in building distributed applications that use Microsoft's Remote Procedure Call (RPC) or Distributed Component Object Model (DCOM) are presented.

An implementation of the WorkCrew multithreaded programming model discussed in Chapter 8 is presented in Appendix A.

This book covers a wide area of multithreading programming problems from theoretical aspects to practical implementation in Win32 C/C++ environment, where all functions related to threads environment are presented. Although this book is originally intended for programmers working in a multithreaded Win32 environment, it describes basic problems that arise in all areas of parallel computing and can be useful to a wider range of programmers.

Marin Golub and Leonardo Jelenković
Faculty of Electrical and Computer Engineering
University of Zagreb
Zagreb
Croatia