In programming language books I've come across, information on multithreading isn't that detailed because you typically don't start with multithreading when learning a new language. In .NET Multithreading, however, author Alan Dennis focuses heavily on the basics of multithreaded programming in the .NET framework. He explains both the functionality of the programming interface that .NET provides and background on how to use threads with the technology. He also helps you understand in which context you should use threads and the risks and benefits of using them. In addition to the core concepts, he also describes closely related problem areas and useful tips on how to succeed in your work.
THOROUGH COVERAGE

*.NET Multithreading* begins with a short introduction to the .NET framework, focusing on threaded applications (Dennis expects you to be familiar with programming in C# or VB.Net). Next, a few background chapters introduce parts of the framework, application domains, and threads, and then the book moves further into the topic's details and concepts.

As concurrency control is one of .NET multithreading's most essential parts, Dennis gives good descriptions of its various aspects over three chapters, covering everything from thread communication to synchronization mechanisms. Over time, you'll hopefully find these 60 pages worth the price of the whole book.

Later chapters cover thread-static and thread-local storage (techniques used in .NET to access data) and closely related concepts, including the delegate pattern, timers, windows forms, interoperability with unmanaged components, and design considerations. Although the delegate pattern introduced in the framework isn't a multithreaded issue only, it's likely to be used in this context. It involves an asynchronous pattern with the possibility of having callback functions.

Dennis also devotes a chapter to exceptions. Using threads introduces special types of exceptions, which he describes complete with code samples. These exceptions illustrate why logging is vital for solving problems encountered during multithreading. The book's final chapter is more like an appendix, ending with a J# example aimed at Java developers.

Down to the details

Throughout the book, Dennis uses fine details to make and support his assertions. For example, in the chapter on thread pooling, Dennis says the maximum number of threads available in the system's thread pool is 25. Specifics like this illustrate the irony of getting the best performance out of an application with the portability that a framework such as .NET provides—a concept that appears throughout the book.

The book also features many code samples that include real-world examples and references. The design chapter in particular features design patterns implemented in a few well-explained code samples and ends with a short, general discussion about performance. Unfortunately, the book provides some code samples in C# and others in VB.Net, which can be a disadvantage if you're not familiar with reading both code syntaxes. However, the downloadable code samples on the publisher's Web site are available in both programming languages and shouldn't be a problem in general. The site also lets you download source code described in the book and communicate with the author in its online forum.
A LONG-LASTING REFERENCE

The first impression I got from the book was that Dennis quickly and concisely gets to the details. He uses language and metaphors that make the book easy to read, with examples, figures, and code samples to help explain concepts. If you're looking for information on general multithreading, with high-performance examples and worst case scenarios, you won't find it in this book.

For anyone who wants to know more about multithreaded computing in .NET, this book, which comes in paperback and electronic versions, can complement online resources. And for more advanced readers, it can complement books dealing more with theory.

CONCLUSION

I recommend .NET Multithreading, as it offers substantial details on the framework. It focuses on .NET from a multithreaded perspective, not the other way around. Normally such information tends to change rapidly over time, but this book will probably remain a good reference for future versions of the framework as well as for readers just gaining experience with .NET.

Mikael Omma is a software consultant at TietoEnator, Telecom & Media. Contact him at mikael.omma@ieee.org.