**Technical Editor:**
**Marcin Paprzycki**
Dept. of Computer Science and Statistics
Univ. of Southern Mississippi
Southern Station 1506
Hattiesburg, MS 39406-1506
m.paprzycki@usm.edu

## Book Reviews

# Structured Development of Parallel Programs

*Reviewed by Andrzej Stachurski, Warsaw University of Technology*

***Structured Development of Parallel Programs***
By Susanna Pelagatti
248 pages
$44.95
Taylor & Francis
London
1997
0-7484-0759-6

*Structured Development of Parallel Programs* presents a structured programming methodology for parallel computations that ensures portability, programmability, and good performance. The book's ultimate goal is to develop a suitable programming language for parallel programming and its compiler. This language is meant to deliver typical parallel constructs (skeletons) and their realizations (templates) on various architectures.

The book's first half presents a critical analysis of the state of the art of parallel software development. It also closely examines several existing approaches to parallel programming, concluding that template-based systems are the best compromise. In this approach, the programmer selects skeletons and their conversion rules, then uses them to build a program. Its performance might not match that of a low-level graph-based approach, but it is predictable and easily ensures programmability and portability.

The book's second half describes the P3L template-based methodology and its realization as the P3L language and its compiler, offering application examples. The author maintains that the template-based system gives rise to accurate performance models for the skeletons library designer as well as for the programmer. The technical and mapping details are left to the skeleton library designer, who can fully exploit specific properties of particular skeletons. The P3L methodology incorporates a small set of basic skeletons and their combination rules. Skeleton selection is based on the analysis of existing approaches. The skeletons reflect typical constructs that parallel program designers use.

The P3L methodology might be a good starting point for developing efficient high-level languages for parallel programming. It suggests how to ensure compromise between performance and portability and programmability. In any case, we should not treat it as something closed and finally established—high-level parallel programming languages continue to develop and improve.

Such high-level languages would let the programmer concentrate less on the details of the machine's architecture and more on the algorithm's design. The lack of high-level languages is one of the major obstacles hampering large, complex software projects and the development of computational algorithms. Currently, the progress of these languages is severely delayed compared to the pure parallel hardware performance. An efficient, high-level language for parallel programming available on computers with parallel processors and on clusters of machines used for distributed computations would be an important tool for people developing general theoretical and application-oriented algorithms.

This book should interest people working on parallel algorithms, but, more importantly, it should interest researchers and software engineers developing languages for parallel computations. It might also be of interest to both undergraduate and graduate computer science students because it does not require any special background. It can supplement material for courses devoted to programming languages and compilation techniques, especially for high-level parallel programming.

Acknowledgments. Developing Parallel Programs -A Discussion of Popular Models. Oracle white paper | may 2016. Table of Contents Introduction Overview An Example of Parallel Programming OpenMP.Â  To aid the development of parallel applications, several programming models have been created. The most frequently used ones are OpenMP [4, 5] for shared memory programming, and MPI [6, 7] for distributed memory programming.Â  Some common data structures are used in implementing the wordcount example in all of the aforementioned frameworks. Good programming practices express the inherent parallelism of the program by using data structures that are portable across multiple frameworks with little modification. Structured Parallel Programming offers the simplest way for developers to learn patterns for high-performance parallel programming. Written by parallel computing experts and industry insiders Michael McCool.Â  Structured Parallel Programming: Patterns for Efficient Computation 1st Edition. by Michael McCool (Author), James Reinders (Author).