Languages for Computational Science and Engineering – Guest Editor's Introduction, IEEE Computational Science and Engineering, Volume 5, Number 2, April-June Issue, 1998

Rudolf Eigenmann L.V. Kale David Padua

Why would you read a theme issue on languages for computational science and engineering?

As a scientist or engineer experienced in computational problem solving you may ask whether Fortran is still the language of choice? Should you start looking into alternatives? Would that be C, C++, or Java?

As a new CS&E student or practitioner you may wonder which languages you should learn for expressing computational problems and implement solutions successfully. What would benefit your career the most?

As a manager of a computational engineering group you need to know what courses you should ask your staff to attend and how this recommendation might change over the next 5 years.

As a computer scientist developing new languages you'd like to see if CS&E could be your application area. What are language-related issues of the future computational application?

Maybe you are interested in the larger issues. What applications are enabled by new languages? Do new languages help thinking about things in new ways? Do changes in programming languages reflect changes in global thinking, just like spoken language does? Do languages drive applications or vice versa?

Then again you may be just curious: Are there any hot topics?

This theme issue gives answers to a few of these questions - all written by leaders in their particular field of languages for applications in computational science and engineering. In addition, the first article surveys programming languages for modern scientific and engineering computations. It briefly presents many of the languages that could not be addressed in detail by the individual papers.

"A Case Study of Fortran in Computational Science and Engineering" by Moreira and Midkiff is devoted to the currently still most-widely used language for CS&E applicatons. The paper targets current and new practitioners. It shows how a simple physics problem can be turned into a Fortran program, explains modern features of Fortran 90 and compares the performance of Fortran versus C++.

Perhaps the biggest issue with software development in general is the low productivity and

high error rate of programs coded in today's standard languages. The CS&E area is no exception. One solution approach is to provide domain-specific languages, which offer higher-level constructs for their application area - usually at the cost of generality. Domain-specific languages are especially interesting for large application areas. One such area is that of discrete event simulations. The paper on "Parallel Languages for Discrete-Event Simulation Models" by Bagrodia, not only represents this area but also the increasingly import field of non-numerical CS&E applications. Many other domain-specific languages exist. The graphical languages of Computer-aided design systems are well-known examples. Domain-specific approaches have also been taken for solving partial differential equations, as they are found in a large number of CS&E applications. Such tools are sometimes refered to as "problem solving environments." A theme issue of the CS&E Magazine was devoted to this topic in Fall 1997 (Volume 4, Number 3).

Two articles point at languages for future CS&E applications. The future computational application is envisioned as one that solves problems of enormous complexity, encompassing multiple disciplines, possibly combining hard-core physics with environmental and socio-economic factors. It will run on world-wide interconnected compute resources and involve design teams in distant locations. The first of the two articles, "Component Architectures for Distributed Scientific Problem Solving" by Gannon, Bramley, et. al., discusses design and interface issues of component-based, object-oriented software. "Language Support for Multidisciplinary Applications" by Methotra, Van Rosendale, and Zima, then describes issues in the design of multidisciplinary applications and presents a new language for expressing and coordinating such large-scope problems.

Evidently, the languages discussed in this theme issue are just a small sample. Many contributions remain unnamed or are presented in less detail than they deserve. One notable new language that has been proposed recently for CS&E applications is called ZPL. It will be described in a full paper in the next issue of the CS&E Magazine.