

## Guest-Editorial

---

# PADDA 2001

This special issue is devoted to programming models, languages, and tools for performance-oriented program development in commercial and scientific environments. The included papers have been written based on presentations given at the workshop PADDA 2001 (Performance-Oriented Application Development for Distributed Architectures – Perspectives for Commercial and Scientific Environments) held at Technische Universität München on April, 19th–20th, 2001. The workshop was funded by KONWIHR (Competence Network for Scientific High Performance Computing in Bavaria, [konwihhr.in.tum.de](http://konwihhr.in.tum.de)).

The goal of the workshop was to identify common interests and techniques for performance-oriented program development in scientific and commercial environments. Distributed architectures currently dominate the field of highly parallel computing. Highly parallel machines can range from clustered shared-memory systems like the Hitachi SR8000 and SGI Origin 2000, to PC-based clusters connected via Myrinet/Beowulf, SCI or Fast Ethernet. In addition, meta-computing applications are executed in parallel on high-performance systems and special purpose systems, which are connected via a wide-area network.

Distributed architectures, based on Internet and mobile computing technologies, are important target architectures in the domain of commercial computing too. Program development is facilitated by distributed object standards such as CORBA, DCOM, and Java RMI. Those standards provide support for handling remote objects in a client-server fashion, but must also ensure certain guarantees on the quality of service. A large number of applications, such as data mining, video on-demand, and e-commerce, require powerful server systems, which are quite similar to the parallel systems used in high performance computing applications.

In both areas performance tuning plays a very important role. For high performance computing both, individual applications as well as co-operating applications in meta-computing environments, have to be tuned to

improve speedup and enable new complex simulations to be undertaken. In commercial data processing performance is crucial to enable processing of huge data set and customer requests. For example, e-commerce applications must support rapid response times to customers, which may become a deciding factor in whether customers remain with a given service provider. Similarly, such electronic businesses may need to sustain a large number of transactions and manage large data sets. A similar requirement is found in emerging applications in scientific computing which make use of distributed visualization and data management.

On one side, there is an uptake of programming standards from the commercial arena into high performance computing since these standards facilitate meta-computing, as well as programmers from universities being trained in these programming styles. On the other hand, a huge experience in performance analysis exists in the area of high performance computing that might be applicable to the commercial arena.

The papers included in the special issue come from both areas, scientific computing and commercial computing. The first five papers are devoted to scientific computing while the other three papers are presenting aspects and techniques for commercial computing.

The first paper written by Tony Hey et al. on *Performance Engineering, PSEs and the Grid* gives an introduction into requirements and standard techniques for performance analysis in scientific environments. It also presents new requirements and techniques in the areas of problem solving environments and grid computing.

The second paper written by José Moreira et al. on *NINJA: Java for High Performance Numerical Computing* discusses the challenges in using Java for scientific programs. It presents the motivation for using Java for numerical applications as well as the current major performance problems in doing so. It presents also recent techniques for improving the performance of numerical codes written in Java.

The third paper written by Eduardo César et al. on *Dynamic Performance Tuning Supported by Program Specification* presents an environment for automatic performance analysis and program tuning for message passing codes. Kappa-Pi analyzes traces from message passing programs, detects performance bottlenecks automatically, and gives recommendations for improving the code. The environment is currently being extended for online-tuning of applications based on abstract program structures, such as SMPD or divide and conquer.

The fourth paper written by Jie Tao et al. on *Memory Access Behavior Analysis of NUMA-based Shared Memory Programs* presents a new monitoring infrastructure for SCI-based NUMA machines. It is based on a low-level hardware monitoring facility in SCI-interfaces and includes software layers and tools that allow the user to investigate the remote access performance in the context of the program's control flow and data structures.

The fifth paper written by Ralf Reussner et al. on *SKaMPI: A Comprehensive Benchmark for Public Benchmarking of MPI* describes a benchmark for MPI programs which incorporates well-accepted mechanisms for ensuring accuracy and reliability. The benchmarking results are maintained in a public performance database with different hardware platforms and MPI implementations. It thus assists users in developing performance portable MPI applications.

The second group consists of three papers presenting aspect of performance-oriented programming in commercial environments. The first paper written by Günther Rackl et al. on *Efficiently Building On-line Tools for Distributed Heterogeneous Environments* presents a methodology for developing on-line tools based on MIMO, a novel middleware monitor. The MIMO environment implements an event-based on-line approach in the context of CORBA applications frequently used in commercial environments.

The second paper written by Arun Iyengar et. al. on

*Architecting Web Sites for High Performance* gives an overview of recent solutions concerning the architectures and the software infrastructures used in building Web site applications. The challenge for such Web sites is the specific combination of high variability in workload characteristics and of high performance demands regarding the service level, scalability, availability, and costs.

The third paper in this group written by Luc Moreau et al. on *Mobile Objects in Java* discusses a novel algorithm for routing method calls. Mobile objects have emerged as a powerful paradigm for distributed applications. Important for the success of that paradigm is the performance of transparent method invocation and the availability of distributed garbage collection. This article compares different implementations based on synthetic benchmarks.

The broad spectrum of topics covered by the papers in this special issue will, hopefully, enable readers to benefit from techniques developed in both areas. New research and development projects should take advantage of similarities of requirements for performance in both areas and thus broaden the opportunities for exploitation of the results. Certainly, the workshop was very successful in steering discussions and triggering new ideas for joint research.

I would like thank the members of the PADDA 2001 program committee, Marios Dikaiakos (University of Cyprus), Wolfgang Gentzsch (SUN Microsystems), John Gurd (University of Manchester), Gabriele Kotsis (Universität Wien), Frank Müller (North Carolina State University), Jim Laden (SUN Microsystems), and Omer F. Rana (University of Wales-Cardiff), for setting up an exiting program and for selecting these excellent papers.

*Michael Gerndt*  
Technische Universität München  
Germany