Journal of Parallel and Distributed Computing **61**, 1827–1841 (2001)
doi:10.1006/jpdc.2001.1767, available online at http://www.idealibrary.com on IDEAL®

## INVITED PAPER

# What Are the Top Ten Most Influential Parallel and Distributed Processing Concepts of the Past Millenium?

Mitchell D. Theys

*Department of Computer Science (MC 152), University of Illinois at Chicago, 851 South Morgan Street, Room 1120, Chicago, Illinois 60607-7053*
E-mail: mtheys@uic.edu

Shoukat Ali

*School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana 47907-1285*

Howard Jay Siegel

*Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, Colorado 80523-1373*

Mani Chandy

*Computer Science Department 256-80, California Institute of Technology, Pasadena, California 91125*

Kai Hwang

*Department of Electrical Engineering Systems, University of Southern California, EEB#200C, 3740 McClintock Avenue, Los Angeles, California 90089-2562*

Ken Kennedy

*Rice University, HiPerSoft-MS41, 6100 Main Street, Houston, Texas 77005-1892*

Lui Sha

*University of Illinois at Urbana–Champaign, 1304 West Springfield Avenue, Urbana, Illinois 61801*

Kang G. Shin

*Real-Time Computing Laboratory, Department of Electrical and Computing Science, University of Michigan, 1301 Beal Avenue, Ann Arbor, Michigan 48109-2122*

Marc Snir

*IBM T. J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598*

Larry Snyder

*Department of Computer Science and Engineering, University of Washington, Box 352350, Seattle, Washington 98195-2350*

and

Thomas Sterling

*Center of Advanced Computing Research, NASA Jet Propulsion Laboratory, M/C 158-79, Pasadena, California 91125*

Accepted April 6, 2001

This is a report on a panel titled "What are the top ten most influential parallel and distributed processing concepts of the last millennium?" that was held at the IEEE Computer Society sponsored "14th International Parallel and Distributed Processing Symposium (IPDPS 2000)." The panelists were chosen to represent a variety of perspectives and technical areas. After the panelists had presented their choices for the top ten, an open discussion was held among the audience and panelists. At the end of the discussion, a ballot was distributed for the audience to vote on the top ten concepts (in arbitrary order). The voting identified the following ten most influential parallel and distributed processing concepts of the last millennium: (1) Amdahl's law and scalability, (2) Arpanet and Internet, (3) pipelining, (4) divide and conquer approach, (5) multiprogramming, (6) synchronization (including semaphores), (7) load balancing, (8) message passing and packet switching, (9) cluster computing, and (10) multithreaded (lightweight) program execution.

## 1. INTRODUCTION

With the turn of the millennium, many groups have viewed the accomplishments from the last millennium and rated how these advances have affected their members. In this spirit, a panel was organized at the IEEE Computer Society sponsored "14th International Parallel and Distributed Processing Symposium (IPDPS 2000)." The panel of senior computer scientists and computer engineers was convened on Tuesday, May 2, 2000, and was titled "What are the top ten most influential parallel and distributed processing concepts of the last millennium?"

Howard Jay Siegel, professor in the Department of Electrical and Computer Engineering, Colorado State University, organized and moderated the panel. The eight panelists listed below were chosen to represent a variety of perspectives and technical areas. Kai Hwang, professor of Electrical Engineering and Computer Science and Director of Internet and Cluster Computing Laboratory at the University of Southern California, specializes in computer architecture, digital arithmetic, and parallel processing, and is presently investigating Internet and cluster computing. Kang G. Shin, professor of Computer Science and Engineering, and the

Founding Director of the Real-Time Computing Laboratory (RTCL) in the Electrical Engineering and Computer Science Department at the University of Michigan, concentrates on real-time computing concerns, including real-time operating systems, middleware services, and real-time applications (multimedia, electronic commerce, telecommunication, etc.). Ken Kennedy, the Ann and John Doerr Professor of Computational Engineering and Director of the Center for Research on Parallel Computation (CRPC) at Rice University, explores architecture independent programming support, including compilation techniques and language design. Larry Snyder, professor of Computer Science and Engineering at the University of Washington, Seattle, is the co-inventor of the Chaos Router and has explored various aspects of parallel computing, including architectural and language issues. Lui Sha, professor of Computer Science at the University of Illinois at Urbana–Champaign, researches real-time systems and is recognized for his contributions to real-time computing theory, changing the real-time computing open standards, and for solving the Mars Pathfinder's priority inversion problem while it was on Mars. Mani Chandy, the Simon Ramo Professor and Executive Officer for Computer Science at Caltech, studies concurrent computing and network models of communication and computation performance. Marc Snir, senior manager of research on scalable parallel systems at the IBM T. J. Watson Research Center, is concerned with scalable massively parallel processing systems, computational complexity, parallel algorithms, and parallel architectures, and was a major contributor to the MPI standard. Thomas Sterling, a principal scientist at the NASA Jet Propulsion Laboratory and a faculty associate at Caltech, is a leader in petaflop computing, and is known for his advances in processor-in-memory execution models and Beowulf cluster computing.

The panelists' presentations are summarized in the order in which they were given (alphabetical order by the first name) in Section 2. This is followed by the highlights of the ensuing discussion in Section 3. Finally, Section 4 reports the results of the informal audience survey regarding their top ten concepts.

## 2. PRESENTATIONS

### 2.1. Guidelines

The panel chair, H. J. Siegel, introduced the topic and the panelists and asked each panelist to present his "top ten list" for the most influential parallel and distributed processing concepts. As a few guidelines, the chair stated that the panelists could present less than 10 items, the ordering of the list could be arbitrary, and the panelists were restricted to 10 minute presentations. The audience members were invited to participate in an open discussion and ask questions after the presentations. Siegel also requested that the audience members vote on the top 10 (in arbitrary order) most influential concepts after the open discussion.

Many of the panelists reported less than 10 concepts. In addition, many of the panelists issued a disclaimer that some of their proposed concepts may predate the millennium being considered.

TABLE 1

**Kai Hwang's Top Ten List**

| | |
|---|---|
| Amdahl's Law | PRAM model |
| Dataflow computing | RISC architecture |
| Distributed shared memory machines | Parallel software or programming |
| RAID | Clusters of computers |
| Internet computing | EPIC—explicit parallelism instruction computing |

## 2.2. *Kai Hwang*

Kai Hwang began his presentation by noting that computing was only 50 years old, and as such, he selected his choices from the past 50 years. Hwang's list of the top ten influential concepts is shown in Table 1. Hwang went on to justify why he thought that the concepts given in his list were the most important concepts in the last millennium. He felt that Amdahl's law was an important concept because it set a limit on parallel processing, and that the PRAM (parallel RAM) model was important because it simplified the development of parallel algorithms. He believed that dataflow computing was important because it enabled fine-grain parallelism. Hwang reasoned that the DSM (distributed shared memory) architecture allowed the combination of the shared-memory and the message passing paradigms and was therefore an important concept. He concluded by stating that his list covered both models, e.g., PRAM and RAID (redundant array of inexpensive disks), and architecture advancements, e.g., RISC (reduced instruction set computing) and clusters.

## 2.3. *Kang G. Shin*

Kang Shin began by stating that the problem was far too difficult to answer, and pointed out that neither computer scientists and engineers nor the computers had existed for a thousand years. Shin emphasized that it was the integration of many concepts, rather than many individual concepts, which had developed computing into its present form. He feared that each person's tunnel vision would produce very subjective answers to the posed question. Then Shin presented his list of top ten items (shown in Table 2). He also addressed the concern about who cared about the answers to the panel question, and why they cared. He believed that Wall Street cared about the answers because Internet companies were becoming very rich based on parallel and distributed computing technology. Shin concluded with the suggestion that high performance Internet servers were a good target for parallel and distributed computing.

## 2.4. *Ken Kennedy*

Ken Kennedy began with an analysis of the question posed to the panel. In particular, he focused on the words "influence" and "concept," arguing that a concept to be included among the most influential of the millennium should have deeply

TABLE 2

**Kang Shin's Top Ten List**

| | |
|---|---|
| Arpanet and its descendents (TCP/IP, UDP) | World Wide Web |
| LANs (especially Ethernet) | Concurrent CPU, I/O, and memory |
| E-mail | Multitasking OSs and parallelizing compilers |
| Stored memory, SIMD, MIMD computers | Distributed shared memory |
| Client-server computing | Distributed embedded systems and devices |

influenced the way we think about parallelism. He asserted that such a concept should be both broad and abstract. In Kennedy's view, most of the important abstract parallel concepts arose from analogies in everyday life. Examples include divide and conquer, cloning, load balancing, and synchronization. He argued that even those concepts that do not immediately seem to arise from everyday life (like scalable communication interconnects) could after some thought be linked to analogous concepts outside the world of parallel computation. Kennedy pointed out that this leads to the question: were these concepts influential because they captured the familiar, or were they familiar because they were uniquely powerful concepts? He remarked that when the panel had disposed of this issue, perhaps it could finally resolve the primacy of the chicken or the egg. Kennedy's top ten list is presented in Table 3.

## 2.5. *Larry Snyder*

Larry Snyder began his presentation by discussing a linear plot (Fig. 1) of the millennium's advances in parallel processing. Snyder pointed out that the plot showed that the advances were clustered near the 2000 end of the timeline, even when the graph was converted to a log time scale (Fig. 2). He felt that even though the advances were clustered at one point, the important parallel and distributed computing concepts were more evenly distributed. He cautioned that we must interpret parallel ideas more broadly to get an even distribution of ideas from the past millennium.

Starting in a light tone, Snyder began his broadening of the traditional parallel concepts by interpreting the Parthenon as a parallel architecture, by contending that the first instance of multithreading was a programmable loom with multiple

TABLE 3

**Ken Kennedy's Top Ten List**

| | |
|---|---|
| Divide and conquer | Scalable interconnect |
| Multiprogramming | Pipelining |
| Synchronization | Multithreading |
| Scalability—Amdahl's law | Cloning of sheep, code, and computations |
| Load balancing | Distributed memory |

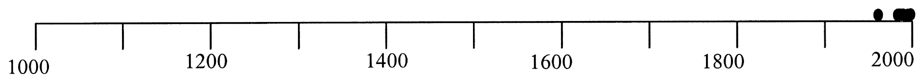

**FIG. 1.** Larry Snyder's plot of millennium's advances in parallel processing using a linear scale.

threads, and by observing that the leaning tower of Pisa was a faulty parallel design. On a more serious note, he stated that we needed to refine the criterion used to find the top ideas of the millennium. He noted that a concept needed to have importance beyond parallel computing to qualify as a concept that had the greatest impact in the last millennium. Using this criterion, Snyder delivered his top five ideas from the second millennium (shown in Table 4).

Snyder thought that "zero" was the second millennium's most important concept. He claimed that "one" was known before the second millennium. Snyder reasoned that the "synthesis that gave the binary alphabet its other half, the coupling of "off" with the ubiquitous "on," the ability to have nothing and know it, to recognize an empty task queue and initialize memory, or even to have memory at all, must be the second millennium's greatest contribution to parallelism and computation generally."

## 2.6. *Lui Sha*

Lui Sha said that his emphasis would be on resource management concepts because he worked in real-time systems. Sha observed that there were two perspectives to consider: (1) application development (e.g., divide and conquer) and (2) foundation and infrastructure. He reasoned that he would present only five key concepts from the last millennium because his work dealt with only one of the two perspectives, foundation and infrastructure. After presenting his top five concepts (shown in Table 5). Sha discussed in greater detail the fifth concept in his list, the concept of priority inversion control (i.e., reducing and bounding the interference of lower priority tasks on higher priority ones). He felt that the priority inversion control was not very well known outside of the real-time computing and multimedia communities. Sha claimed that the standard computer hardware bus arbitration logic and operating systems synchronization mechanisms suffered from unbounded priority inversion until this problem was addressed in the late 1980s to early 1990s. He stressed that this concept had transformed the infrastructure of resource allocation mechanisms and has become part of open hardware and software standards including IEEE POSIX Real-time Extension, Ada 95, IEEE Futurebus+, and all the later standards related to real-time computing such as Real-Time CORBA and Real-Time JAVA.

## 2.7. *Mani Chandy*

Mani Chandy began his presentation with the observation that a system might (1) use concurrency for speed or (2) be inherently concurrent. Chandy claimed that the phrase parallel computing was often used for a system that used concurrency for speed, and that the phrase distributed computing was often used for an

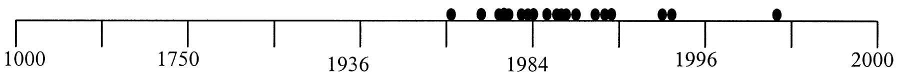

**FIG. 2.** Larry Snyder's plot of millennium's advances in parallel processing using a log scale.

inherently concurrent system. He noted that a parallel computing problem consisted of selecting an algorithm, a programming language, and a machine to execute a given computational task with adequate speed. He claimed that designers of parallel computing systems did not have to use concurrency; they might prefer to use sequential algorithms provided they executed with adequate speed. Chandy stated that most of the concepts that he considered important were in distributed computing because, in his view, inherent concurrency would play a much more critical role in the next millennium than discretionary concurrency would. Chandy further noted that distributed computing dealt with collaboration among collections of people, information appliances, and objects. He felt that the designers of those infrastructures could not choose to eliminate concurrency from consideration because concurrency was part of the specification. He remarked that we needed to examine if concurrency was part of the problem or part of the solution.

Chandy felt that the advances in parallel computing were disappointing. He reasoned that someone like Newton could be taught the last millennium's advancements in parallel computing in a one-month long tutorial, but that modern chemistry or modern physics could not be taught in the same time period. He asked why parallel computing had not advanced as much as modern chemistry or physics had.

Chandy then presented his list of influential concepts (shown in Table 6). He also mentioned several disappointments in distributed computing (shown in Table 7). He commented that although our efforts in advancing the economy might be commendable, not much of the parallel computing work was truly revolutionary. Chandy concluded by asserting that if the advances in parallel computing were truly revolutionary, it would take Newton more than a month-long tutorial to understand them.

## *2.8. Marc Snir*

Marc Snir started his discussion by posing the following question. "Twenty years ago we had distributed systems with 10's to 100's of loosely connected, full-function computers, parallel systems with 100's of tightly connected, sub-function computers, and massively parallel systems with 1000's or 1,000,000's of tightly coupled,

**TABLE 4**

**Larry Snyder's Top Five List**

| | |
|---|---|
| Broadcast communication | Clocking |
| Point-to-point communication | Pipelining |
| Zero | |

TABLE 5

**Lui Sha's Top Five List**

| | |
|---|---|
| Mutual exclusion and synchronization | State feedback |
| Rate control | Weighted fair queuing |
| Priority inversion control | |

sub-function computers. Today we have a distributed system called the World Wide Web, with 100,000's to 1,000,000's of interacting computers, a 64-way shared memory multiprocessor is considered a large-scale parallel system, and a cluster of 1,500 4-way SMPs is considered to be a massively parallel system. Have we gone backwards?" Snir asked the audience if the parallel computing community had stopped dreaming and if massive parallelism was dead. Then he announced that he had a plan (in contrast to a dream) for massively parallel computing. Snir's plan was to build a one petaflop massively parallel system comprising more than 10 million execution threads. He reasoned that because a highly parallel computer could beat the best human chess player, massively parallel computers may be the way to intelligent computing. He pointed to the brain as an example of a massively parallel system that had worked well (at $10^{19}$ operations per second). He claimed that the parallel computing community could achieve (small) brain performance on a supercomputer in less than ten years.

Snir also felt that neural networks could serve as a feasible model for the massively parallel systems. He claimed that small neural networks could learn to perform specific tasks very efficiently, and that (infinite) neural networks could serve as universal compute devices. As other examples of massively parallel systems that had worked well, he offered the (human) body and cellular automata. Snir's top five list is given in Table 8.

## *2.9. Thomas Sterling*

Thomas Sterling began by expressing that his understanding of the panel question was different from that of other panelists' understanding. Sterling felt that the answer to the panel question should involve both the enabler and the constraint on parallelism—the market.

He then presented his list of concepts (shown in Table 9). After presenting the list, Sterling mentioned that our ideas of parallelism had parallels in the real world, and that we could benefit from those analogues in our own computing universe. His examples of real-world parallelism included military troops as an example of

TABLE 6

**Mani Chandy's Top Four List**

| | |
|---|---|
| Problem partitioning | Synchronization |
| Domain decomposition | Determinism |

**TABLE 7**

**Mani Chandy's Top Seven Distributed Computing Disappointments List**

| | |
|---|---|
| Massively distributed peer to peer systems | Formal methods |
| Authentication | Mobile agents |
| Standards | Tiered systems |
| Encryption | |

SIMD, galactic clusters as multitasking, and e-mail as message passing. Sterling felt that there were other parallelism paradigms in the real world that we were not fully exploiting. He cited power generation, transmission, and distribution; gravitational systems; and speciation and evolution as examples of those paradigms. For the case of a gravitational system like the solar system, Sterling pointed out that each planet had its own measure of the system state (i.e., the gravitational field through superposition of the contributions of the gravity forces from other planets). Furthermore, the net force was calculated within the space at each point, and the change in acceleration, velocity, and position was made for each planet at the same time. He posed the question: how could we model the amounts of parallelism available in these systems? He also provided a list of constraints that have hindered parallel computer design. This list is given in Table 10.

Sterling concluded by discussing "hyper-super-massive-parallelism." He thought the concept of massive parallelism was used too soon. He suggested that massive parallelism should have the following attributes: relaxation (finding a global solution in distributed systems through local interactions using successive approximations), ensemble (different elements of a distributed system performing the computation simultaneously with their own version of the global data), stochastic, nondeterministic, and "localized action yields global behavior." Elaborating more on the last attribute, he felt that the emergence of global behavior and rules from local and parallel behavior and rules (e.g., as in Conway's game of life) was a very important concept, and a good example of synergy or symbiosis. Sterling asserted that the universe was the ultimate constraint-based parallel computer, with the program being the collection of physical laws and the input data being the initial conditions at the start of the universe.

## 3. GROUP DISCUSSION

After the panelists presented their ideas, the floor was opened for discussion. The first question asked of the panelists was "What is required in the next millennium to manage these massively parallel systems of the future?" Snir answered by pointing out that one had to let go of the deterministic approaches to understanding systems, and that one had to examine stochastic and continuous modeling to analyze large systems. Sterling claimed that programming constructs of the future would dispense of the program counter, and focus on local action. This would then

TABLE 8

**Marc Snir's Top Five List**

| | |
|---|---|
| Neural networks | Machine chess/artificial intelligence |
| Digital simulations of continuous events | Packet switching |
| Cellular automata | |

merge the ideas of program flow and data structure into the same semantic construct, which would then allow fine grain parallelism to be used independently of particular resources. Chandy thought that perhaps the decision to build a digital system was a step backward. But he claimed that this backward step was taken because customers would not buy a computer that sometimes ran slow and sometimes ran fast, and that the marketplace wanted exactly deterministic behavior.

Hwang commented that massive parallelism would not lead to intelligence. He felt that the only way to gain intelligence was to create systems that were more biological. Chandy noted that computer scientists built computers based upon their observations of nature, whereas chemists investigated outside of their knowledge base and designed systems that were not seen in nature. Sha remarked that although most systems found in nature were analog, computer scientists had moved to digital systems because digital systems were deterministic and predictable. Snir said that when one dealt with a very large system, one could not attempt to analyze the system at a low level. He gave the example of the Internet as a large system, where one would not look at individual packets entering the Internet. Sha interjected that the Internet was not trying to solve a single large problem; instead it was trying to solve many different problems.

An audience member stated that in the current marketplace, influence and popularity determined which software was used most often, not worthiness. As a result, users were stuck with software that was unreliable and resulted in problems with performance and productivity. He then asked the panelists if that state existed because good ideas did not have enough influence, or because there were not enough good ideas. Chandy began the discussion by asking if concurrency was part of the problem or part of the solution. He felt that concurrency was part of the problem with the World Wide web. Chandy claimed that for intelligence, concurrency was not part of the problem. He also thought that intelligence would be found with a very fast machine, not necessarily a parallel machine. Kennedy

TABLE 9

**Thomas Sterling's Top Ten List**

| | |
|---|---|
| Bit level parallelism | Multitasking |
| Communicating sequential processors | Fine rain pipelining |
| SIMD/SPMD | Dataflow |
| Systolic systems | Multithreading |
| Cellular automata | Futures |

**TABLE 10**

**Thomas Sterling's Parallelism Implementation Constraints**

| | |
|---|---|
| Cost of devices | Limited resources |
| Overhead and temporal costs | Mass market |
| Software mindset/models | |

continued by looking at parallel software and how the market had influenced it. He thought that good ideas had been developed that did not have the desirable influence due to the market. Sterling stated that there were two points that should be considered: choice of program and choice of management. He thought that, currently, the same people handled both of these areas, and that should be done differently in the future.

Another audience member thought that a historian would do a statistical analysis on how often concepts appeared most often in the literature to decide how important a given concept was. In this regard, he thought that Flynn's taxonomy was one of the most cited concepts in the literature, and deserved an honorable mention. The panel chair, however, said that many students now in universities had never

**TABLE 11**

**The Ballot for the Audience Survey**

| | |
|---|---|
| Adaptive distributed systems | Load balancing |
| Amdahl's Law and scalability | Market |
| Arpanet and subsequent Internets (TCP/UDP/IP) | Message passing and packet switching |
| | MPI |
| Basic data structures | Multiprogramming |
| Big O and asymtotics | Multithreaded (lightweight) program execution |
| Cache coherence | NP-completeness and reductions |
| Cellular automata (systolic arrays) | Neural networks |
| Client-server computing | One-sided (asynchronous) communication |
| Cloning | Optical fiber technology and wireless technology |
| Cluster computing | |
| Communicating sequential processes (aka message passing) | parallel programming |
| | Parallel search |
| Computer chess and Deep Blue | Pipelining including vector operations, overlapping memory bank access, and wormhole routing |
| Data distribution | |
| Data parallelism, including array SIMD and some vector | |
| | PRAM |
| Dataflow with I-structures and futures | Priority inversion |
| Depth-first search and applications | PVM |
| Digital simulation | Quality of service |

**TABLE 11**—*Continued*

| | |
|---|---|
| Distributed memory systems (not shared) | RAID |
| Distributed shared memory systems | RISC microprocessor |
| Divide and conquer | Remote procedure call |
| Encryption and authentication algorithms | RSA, Merkle, and Diffie-Hellman for public-key cryptography |
| EPIC paradigm | Scalable communications interconnect |
| Ethernet | Scalable database servers |
| FFT algorithm | Synchronization (including semaphores) |
| Ford/Fulkerson for maxflow/mincut algorithm | Transaction systems |
| Fork-join | Understanding composition (of complex systems from simpler ones) and state |
| Full/empty bits | URLs |
| Futures | Vector architectures |
| Incompleteness and incomputability | Web |
| Interleaving model | Weighted fair queuing |
| Java | XML |
| Job stream parallelism | Quantum computing |
| Gauss for his text and work on factoring | Von Neuman architecture |
| Linear programming | Zero |

heard of "SIMD" or "MIMD" acronyms used in Flynn's taxonomy. Kennedy suggested there could be many ways one could decide upon the top ten concepts of the millennium. He said that one could look for the top ten most cited papers, and then determine the concepts that were in those papers; or the audience could vote on the individual concepts; or one could look for successful systems that were based on some subset of the ideas that were being considered for the top ten.

**TABLE 12**

**The Top Ten Most Influential Concepts of the Last Millenium**

| Number of votes | Concept |
|---|---|
| 96 | Amdahl's Law and scalability |
| 95 | Arpanet and Internet |
| 79 | Pipelining |
| 63 | Divide and conquer |
| 53 | Multiprogramming |
| 50 | Synchronization (including semaphores) |
| 42 | Load balancing |
| 42 | Message passing and packet switching |
| 40 | Cluster computing |
| 39 | Multithreaded (lightweight) program execution |

## 4. VOTING

Prior to the panel discussion, a ballot was prepared for the distribution to the audience members. This ballot listed all of the suggestions received from panelists prior to the panel and was slightly modified and expanded at the end of the panel based on comments from the panelists and audience. The ballot, edited slightly for clarity, is shown in Table 11. The audience (and panelists) then voted for up to 10 top choices (in an arbitrary order). A total of 139 ballots were returned. The top ten most voted concepts, along with the number of votes cast, are shown in Table 12.

## ACKNOWLEDGMENTS

The authors thank the IPDPS Steering Committee for organizing the panel topic, Chip Weems for suggesting the audience ballot, and Sally Jelinek and Susamma Barua for implementing the ballot.

---

MITCHELL D. THEYS received a Ph.D. in electrical and computer engineering in 1999 from Purdue University. In addition, he received an M.S. in electrical engineering in 1996, and a B.S. in computer and electrical engineering in 1993, both from Purdue University. He is currently an assistant professor at the University of Illinois at Chicago. His current research interests include: distributed computing, heterogeneous computing, parallel processing, VLSI design, and computer architecture. Prof. Theys has published several journal papers, and also had papers at conferences such as the International Conference on Parallel Processing, and the Heterogeneous Computing Workshop. He has received support from Defense Advanced Research Projects Agency (DARPA), Intel, Microsoft, and the Armed Forces Communications and Electronics Association (AFCEA). He is a member of the IEEE, IEEE Computer Society, Eta Kappa Nu, and Tau Beta Pi. (Department of Computer Science (MC 152), University of Illinois at Chicago, 851 S. Morgan St. RM 1120, Chicago, IL 60607-7053; email: mtheys@uic. edu)

SHOUKAT ALI is pursuing a Ph.D. degree from the School of Electrical and Computer Engineering at Purdue University, where he is currently a DARPA-supported Research Assistant. Shoukat received his M.S. degree in electrical engineering from Purdue University in 1999, and his B.S. degree in electrical and electronic engineering from the University of Engineering and Technology, Lahore, Pakistan in 1996. His main research topic is dynamic mapping of meta-tasks in heterogeneous computing systems. He has held teaching positions at Aitchison College and the Keynesian Institute of Management and Sciences, both in Lahore, Pakistan. He was also a Teaching Assistant at Purdue. His research interests include computer architecture, parallel computing, and heterogeneous computing. (School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285; email: alis@purdue.edu)

HOWARD JAY SIEGEL received B.S.E.E and B.S. Management degrees from MIT (1972), and the M.A. (1974), M.S.E. (1974), and Ph.D. (1977) degrees in computer science from Princeton University. He is the Abell Distinguished Professor of Electrical and Computer Engineering and a Professor of Computer Science at Colorado State University. From 1976 to 2001, he was a Professor in the school of Electrical and Computer Engineering at Purdue University. His research interests include heterogeneous parallel and distributed computing, distributed communications networks, parallel processing, and interconnection networks for parallel machines. He is an IEEE Fellow and an ACM Fellow. He has co-authored over 280 technical papers, was a Coeditor- in-Chief of the Journal of Parallel and Distributed Computing, and served on the Editorial Boards of the IEEE Transactions on Parallel and Distributed Systems and the IEEE Transactions on Computers. (Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523; email: hj@colostate.edu)

MANI CHANDY received the B.Tech degree from the Indian Institute of Technology, Madras (1965), the M.S. degree from the Polytechnic University of New York (1966), and the Ph.D. degree from MIT (1969), all in electrical engineering. He is the Simon Ramo Professor of Computer Science at the California Institute of Technology. He is currently on leave of absence from Caltech, working for iSpheres in Oakland, CA. He held the Assistant to Full Professor and Chair positions at the Department of Computer Science, University of Texas at Austin, from 1970 to 1987. He has also been a staff member of the IBM Scientific Center. His research interests include distributed systems for crisis management. The problem is that of developing a software platform that can be customized rapidly to provide the infrastructure for decision support to task forces dealing with unfolding crises such as hurricanes and earthquakes. Very similar problems also arise in health care, the finance industry and in the supply chain. (Computer Science 256-80, California Institute of Technology, Pasadena, CA 91125; email: mani@cs.caltech.edu)

KAI HWANG received the Ph.D. from University of California at Berkeley. He is a Professor and Director of Internet and Cluster Computing Lab at the University of Southern California. An IEEE Fellow, he specializes in computer architecture, digital arithmetic, and parallel processing. He has published extensively in these areas. He is the founding Editor-in-Chief of the Journal of Parallel and Distributed Computing. His latest books, Scalable Parallel Computing (McGraw–Hill 1998) and Advanced Computer Architecture (McGraw–Hill 1993) have been widely used in computer science and engineering courses. He has served as a consultant and lectured worldwide for IBM, Intel, Fujitsu, MIT Lincoln Lab, Japan's ETL, GMN in Germany, and CERN Computing School. He has chaired the IEEE/ACM ARITH-7, ICPP-86, IPPS-96, and HPCA-97 and will chair the IEEE Cluster 2001. Presently, he leads a cluster research group at USC developing web security architecture and middleware support for federated B2B or B2G E-commerce and collaborative engineering design. (Department of Electrical Engineering Systems, EEB#200C, 3740 McClintock Ave., University of Southern California, Los Angeles, CA 90089-2562; email: kaihwang@usc.edu)

KEN KENNEDY received his B.A. from Rice University in 1967 and his M.S. and Ph.D. from New York University in 1969 and 1971, respectively. He is the Ann and John Doerr Professor of Computational Engineering and Director of the Center for High Performance Software Research at Rice University. Prof. Kennedy's research interests include parallel computing, scientific programming environments, and optimization of compiled code. He has published over 150 technical articles and supervised 34 Ph.D. dissertations on programming support software for high-performance computer systems. He is a Fellow of the AAAS, ACM, and IEEE and is a member of the National Academy of Engineering. For his research accomplishments, he received the IEEE W. Wallace McDowell Award in 1995 and the ACM Programming Languages Achievement Award in 1999. (Rice University, HiPerSoft-MS41, 6100 Main Street, Houston, TX 77005-1892; email: ken@cs.rice.edu)

LUI SHA obtained his Ph.D. from Carnegie Mellon University in 1985. He is currently a professor in the Computer Science Department of the University of Illinois at Urbana-Champaign, the chair of IEEE Real Time Systems Technical Committee, and an associate editor for the Journal of Real Time Systems. He is interested in flexible and reliable real time computing systems. He was elected to be an IEEE Fellow "for technical leadership and research contributions, which enabled the transformation of real-time computing practice from an ad hoc process to an engineering process based on analytic methods." (2107 DCL, 1304 West Springfield Avenue, Urbana, IL 61801; email: lrs@uiuc.edu)

KANG G. SHIN received the B.S. degree in Electronics Engineering from Seoul National University, Korea (1970), and the Ph.D. degree in Electrical Engineering from Cornell University (1978). He is a Professor and Founding Director of the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, the University of Michigan, Ann Arbor, Michigan. His current research focuses on Quality of Service sensitive computing and networking with emphases on timeliness and dependability and their application to telecommunication, multimedia systems, and embedded systems. He has supervised the completion of 41 Ph.D. theses, and (co-authored over 600 technical articles and a textbook "Real-Time Systems" (with C. M. Krishna), McGraw–Hill (1997). He has

received numerous awards including the Outstanding IEEE Transactions on Automatic Control Paper Award (1987), Best Paper Awards from IEEE RTAS (1997) and USENIX Technical Conference (2000), Research Excellence Award (1989) and Outstanding Achievement Award (1999) from University of Michigan. (Real-Time Computing Laboratory, Department of Electrical Engineering & Computer Science, The University of Michigan, 1301 Beal Avenue, Ann Arbor, MI 48109-2122; email: kgshin@eecs.umich.edu)

MARC SNIR received a Ph.D. in Mathematics, from the Hebrew University of Jerusalem, in 1979. He is a senior manager at the IBM T. J. Watson Research Center, where he leads research on the Blue Gene massively parallel system. He joined the IBM T. J. Watson Research Center in 1986 and headed research that led to the line of high performance IBM SP systems. Dr. Snir is a member of the IBM Academy of Technology, an ACM Fellow, and an IEEE Fellow. (IBM T. J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598; URL: http://www.research.ibm.com/people/s/snir; email: snir@Watson.ibm.com)

LAWRENCE SNYDER received his B.A. from the University of Iowa and his Ph.D. from Carnegie-Mellon University. He is a professor of computer science at the University of Washington. He joined the faculty in 1983 after serving on the faculties of Yale and Purdue universities. His research has ranged from proofs of the undecidability of properties of programs to the design and development of a 32-bit single chip (CMOS) microprocessor. Recently he has been principal investigator on the Chaos Network Routing Project and the ZPL Parallel Language Project. Prof. Snyder chaired the National Science Foundation Advisory Committee for the Division of Computer Research and participates on numerous national advisory committees on future research directions in parallel computation and computer science policy. He chaired the NRC committee that issued a major report on academic careers in experimental computer science ("Academic Careers for Experimental Computer Scientists and Engineer,") and a major report on computer literacy ("Being Fluent with Information Technology"). He is a Fellow of the IEEE and ACM. (Department of Computer Science and Engineering, Box 352350, University of Washington, Seattle WA 98195-2350; email: snyder@cs.washington.edu)

THOMAS STERLING received his Ph.D. in Electrical Engineering from MIT as a Hertz Fellow in 1984. He is a Faculty Associate at the California Institute of Technology Center for Advanced Computing Research and a Principal Scientist at the NASA Jet Propulsion Laboratory. Dr. Sterling has engaged in research in high performance parallel computer architecture at the Harris Corporation, the IDA Supercomputing Research Center, and the NASA Goddard Space Flight Center, and taught computer architecture at the University of Maryland, College Park. He has made significant contributions in the field of cluster computing leading the initial Beowulf project and in petaflops-scale computing as principal investigator of the interdisciplinary HTMT research project. He has served on the President's Information Technology Advisory Committee sub-panels on high performance computing and open source software, is the author of three books including the recent popular MIT Press book "How to Build a Beowulf," and holds six patents. Dr. Sterling currently is the principal investigator of the Gilgamesh project to develop an advanced multithreaded processor-in-memory architecture for spaceborne high performance computing. (Center for Advanced Computing Research, M/C 158-79, Pasadena, CA 91125; email: tron@cacr.caltech.edu).