

Scanning the Issue

Special Issue on Grid Computing

I. INTRODUCTION

The growth of the Internet, along with the availability of powerful computers and high-speed networks as low-cost commodity components, is changing the way scientists and engineers do computing and is also changing how society in general manages information and information services. These new technologies have enabled the clustering of a wide variety of geographically distributed resources, such as supercomputers, storage systems, data sources, instruments, and special devices and services, which can then be used as a unified resource. Furthermore, they have enabled seamless access to and interaction among these distributed resources, services, applications, and data. The new paradigm that has evolved is popularly termed as “Grid” computing. Grid computing and the utilization of the global Grid infrastructure have presented significant challenges at all levels, including conceptual and implementation models, application formulation and development, programming systems, infrastructures and services, resource management, networking, and security and have led to the development of a global research community.

II. GRID COMPUTING—AN EVOLVING VISION

The Grid vision has been described as a world in which computational power (resources, services, data) is as readily available as electrical power and other utilities, in which computational services make this power available to users with differing levels of expertise in diverse areas, and in which these services can interact to perform specified tasks efficiently and securely with minimal human intervention.

Driven by revolutions in science and business and fueled by exponential advances in computing, communication, and storage technologies, Grid computing is rapidly emerging as the dominant paradigm for wide area distributed computing. Its goal is to provide a service-oriented infrastructure that leverages standardized protocols and services to enable pervasive access to and coordinated sharing of geographically distributed hardware, software, and information resources. The Grid community and the Global Grid Forum¹ are in-

vesting considerable effort in developing and deploying standard protocols and services that enable seamless and secure discovery, access to, and interactions among resources, services, and applications. This potential for seamless aggregation, integration, and interactions has also made it possible for scientists and engineers to conceive a new generation of applications that enable realistic investigation of complex scientific and engineering problems.

This current vision of Grid computing certainly did not happen overnight. In what follows, we trace the evolution of Grid computing from its roots in parallel and distributed computing to its current state and emerging trends and visions.

A. *Origins of the Grid*

While the concept of a “computing utility” providing “continuous operation analogous to power and telephone” can be traced back to the 1960s and the Multics Project [4], the origins of the current Grid revolution can be traced to the late 1980s and early 1990s and the tremendous amounts of research being done on parallel programming and distributed systems. Parallel computers in a variety of architectures had become commercially available, and networking hardware and software were becoming more widely deployed. To effectively program these new parallel machines, a long list of parallel programming languages and tools were being developed and evaluated [14]. This list included Linda, Concurrent Prolog, BSP, Occam, Programming Composition Notion, Fortran-D, Compositional C++, pC++, Mentat, Nexus, lightweight threads, and the Parallel Virtual Machine, to name just a few.

To developers and practitioners using these new tools, it soon became obvious that computer networks would allow groups of machines to be used together by one parallel code. Networks of workstations (NOWs) were in regular use for parallel computation. Besides just homogeneous sets of machines, it was also possible to use heterogeneous sets of machines. Indeed, networks had already given rise to the notion of *distributed computing*. Using whatever programming means were available, work was being done on fundamental concepts such as algorithms for consensus, synchronization, and distributed termination detection. Systems such as the Distributed Computing Environment (DCE) were built to facilitate the use of groups of machines, albeit in relatively

Digital Object Identifier 10.1109/JPROC.2004.842744

¹[Online]. Available: <http://www.ggf.org/>

static, well-defined, closed configurations². Similarly, the Common Object Request Broker Architecture (CORBA) managed distributed systems by providing an object-oriented, client-side application program interface (API) that could access other objects through an Object Request Broker (ORB)³.

Since different codes could be run on different machines, yet still be considered a part of the same application, it was possible to achieve a distributed end-to-end system capability, such as data ingest, processing, and visualization/post-processing. This was sometimes called *metacomputing* [3]. Even then, the analogy between this style of computing and the *electrical power grid* was clear:

The Metacomputer is similar to an electricity grid. When you turn on your light, you don't care where the power comes from; you just want the light to come on. The same is true for computer users. They want their job to run on the best possible machine and they really don't care how that gets done [16].

Of course, the development of programming languages and tools that attempted to transparently harness "arbitrary" sets of machines served to highlight a host of issues and challenges. First, there was no real way to discover what machines were available. In some cases, a hand-coded local resource file defined the "universe" of machines that a parallel/distributed application knew about. Binaries had to be prestaged by hand to a file system local to the remote machines on a well-known path. Contacting any of these machines and starting tasks was typically done using basic UNIX services such as *rsh* and managed using *.rhost* files. Needless to say, security was virtually nonexistent.

Furthermore, these new programming systems were focusing on new and novel syntax for expressing and managing the semantics of parallel and distributed computation. Once up and running, an application code had no idea of the state of its execution environment or what process or network performance it was getting, unless it did passive self-monitoring or deployed its own monitoring infrastructure. When a process, machine, or network connection failed, it was up to the user to diagnose what happened. (When you are using an experimental compiler and *printf()* ceases to work, you know you are in big trouble.) Needless to say, fault tolerance was virtually nonexistent.

1995 was a watershed year. Under the leadership of the National Center for Supercomputing Applications, Urbana-Champaign, IL; Argonne National Laboratory (ANL), Argonne, IL; the San Diego Supercomputing Center, San Diego, CA; and Sandia National Laboratory, Albuquerque, NM, the International Wide-Area Year (I-WAY) was hosted at Supercomputing'95⁴ in San Diego, CA [5]. The I-WAY sought to demonstrate the potential of distributed, virtual supercomputing by hosting over 60 applications on a national testbed [11], [12]. This testbed was cobbled together

²The Open Software Foundation released the source code for DCE 1.0 in early 1992. [Online]. Available: <http://www.opengroup.org/dce/>

³The CORBA 1.0 specification was released in October 1991. [Online]. Available: <http://www.corba.org/>

⁴[Online]. Available: <http://www.supercomp.org/>

in a matter of months across many different institutions and included relatively primitive tools for the scheduling of machines for different applications and for security for their access [7].

The trials and tribulations of such an arduous demonstration paid off, since it crystallized for a much broader segment of the scientific community what was possible and what needed to be done [15]. In early 1996, the Globus Project officially got under way after being proposed to the Advanced Research Projects Agency (ARPA) in November 1994. The process and communication middleware system called *Nexus* [9] was originally built by ANL to essentially be a compiler target and provide *remote service requests* across heterogeneous machines for application codes written in a higher-level language. The goal of the Globus project [1] was to build a *global Nexus* that would provide support for resource discovery, resource composition, data access, authentication, authorization, etc. The first Globus applications were demonstrated at Supercomputing'97⁵.

Globus was by no means alone in this arena. During this same time period, the Legion project [10] was generalizing the concepts developed for Mentat into the notion of a "global operating system." The Condor project [6] was already harvesting cycles from the growing number of desktop machines that a typical institution was now deploying. The UNiform Interface to COmputing REsources (UNICORE) project [2] was started in Germany in 1997.

Backing up in time to 1995, Smarr and Catlett at the National Center for Supercomputing Applications (NCSA) had constructed a cluster of SGI Power Challenge parallel computers that they called the *Power Challenge Array*. They envisioned a distributed metacomputer of these machines at partner sites around the country that they intended to call the *SGI Power Grid*. When the National Science Foundation (NSF) supercomputing centers were recomputed in 1996, researchers at the consortium of NCSA; University of Illinois at Urbana-Champaign; Rice University, Houston, TX; Indiana University, Bloomington; ANL, and the University of Illinois, Chicago, decided to expand on the Power Grid concept. In their proposal to the NSF, they stated:

[Our vision] is the integration of many computational, visualization and information resources into a coherent infrastructure We refer to the integrated resources as the "Power Grid" or simply the Grid.

After this, the term *Grid* rapidly replaced the use of *meta-computer*. At Supercomputing'97, the Globus demonstration testbed was called a *computational Grid*. By 1998, the term *Grid computing* was firmly established with the publication of *The Grid: Blueprint for a New Computing Infrastructure* by Foster and Kesselman [8].

During this time, the interest and momentum of Grid computing was rapidly growing in both academia and industry. This was facilitated (in no small part) by the explosive growth and adoption of the World Wide Web by all segments of science, industry, commerce, and society. The precedent of the World Wide Web made it very easy for a large number

⁵[Online]. Available: <http://www.supercomp.org/>

of people to conceptually extrapolate the serving of Web pages to the discovery and management of computing resources, in general, distributed across a Grid. This growing interest prompted the formation of a Grid Forum to produce and promote standards that industry could build products to. The Grid Forum had its first meeting in June 1999 at NASA Ames while similar efforts were getting started in Europe and the Asia-Pacific region. All of these efforts merged into the Global Grid Forum, which had its first meeting in March 2001 at the Amsterdam Science and Technology Center.

B. Current Goals and Vision

This evolution of the Grid has led to the current vision of Grid computing—a vision of uniform and controlled access to computing resources, seamless global aggregation of resources enabling seamless composition of services, and leading to autonomic self-managing behaviors. This vision applies to all manner and scale of computing resources—from personal digital assistants (PDAs), to enterprise Grids, to an open-ended, global-scale Grid environment, i.e., *the Grid*.

1) *Seamless Aggregation of Resources and Services*: Early Grid computing efforts focused on aggregating geographically distributed resources spanning multiple administrative domains, and this remains a key goal. Aggregation included both the aggregation of capacity (e.g., clustering of individual systems to increase computational power or storage capacity) as well as the aggregation of capability (e.g., combining a specialized instrument with a large storage system and a computing cluster). Key capabilities to enable such aggregation included protocols and mechanisms to secure discovery, access to and aggregation of resources for the realization of *virtual organizations* and the development of applications that can exploit such an aggregated execution environment.

These goals have motivated the generalization of all Grid resources into *services*. A key driver for this generalization was the emergence of Web Services as a dominant technology in the e-commerce domain. This formulation of Grid computing built on the concept of a *Grid service* as the fundamental abstraction, allowing Grids and Grid applications to consist of dynamically composed services.

2) *Ubiquitous Service-Oriented Architecture*: While Grid computing historically grew from the desire to do distributed, virtual supercomputing, the capabilities needed to accomplish this are actually quite fundamental with a far-reaching, broader impact. The ability to do resource and data discovery along with resource scheduling and management in a secure, scalable, open-ended environment based on *well-known* and *widely adopted services* enables a wide variety of application domains and styles of computation. These fundamental capabilities enable not only distributed supercomputing, but also Internet computing, Web computing, cycle harvesting, peer-to-peer computing, etc. In short, we could refer to this as a *ubiquitous service-oriented architecture*—machines large and small (from wireless PDAs to big iron supercomputers) and services that they

provide could be dynamically combined in a spectrum of virtual organizations according to the needs and requirements of the participants involved.

Such an architecture should also be *language/programming model agnostic* and facilitate *interoperability*. Hence, rather than imposing a particular programming model, it should enable the integration of a wide variety of programming models. For example, rather than imposing an object model, as CORBA does, it should allow a CORBA-based object-oriented system to be composed with another non-object-oriented system, etc. By providing a common set of interoperable Grid-level services, it should be easier to produce an interoperable set of application-level services.

3) *Autonomic Behaviors*: The inherent scale, heterogeneity, dynamism, and nondeterminism of Grids and Grid applications have resulted in complexities that are quickly breaking current paradigms, making both the infrastructure and the applications brittle and insecure. Clearly, there is a need for a fundamental change in how Grids and Grid applications are developed and managed. This is leading researchers to consider alternative paradigms that are based on the strategies used by systems in nature to deal with complexity, dynamism, heterogeneity, and uncertainty. This emerging vision aims at realizing computing systems and applications capable of configuring, managing, interacting, optimizing, securing, and healing themselves with minimum human intervention, and has led to a number of recent research initiatives such as Autonomic Grids, Cognitive Grids, and Semantic Grids.

III. MAKING GRIDS A REALITY

While Grids have come a very long way from the efforts of several labs trying to address thorny, fundamental issues in distributed computing by building research prototypes, Grids still have a very long way to go before they are a practical, *widely deployed* reality. At the current time, several basic Grid tools are stabilizing and many Grid projects, including some very well funded international projects, are deploying sizable Grids. However, one can argue that Grids will not be a practical reality until: 1) there is a core set of Grid services with 2) sufficient reliability that are 3) widely deployed enough to be usable. This is the current challenge for making Grids a reality. The issue is how to make this happen.

A. Expanding the Scale and Scope of Deployment

A number of very large Grid projects are currently underway. Examples include the EU DataGrid project, the NSF TeraGrid project, and the Japanese NaReGI project. Many other smaller projects are currently underway, too, involving just a few institutions in a specific application domain. There are also a number of Grid-like commercial products for cycle harvesting, distributed scheduling, etc. In all these cases, however, the deployment and use of the Grid tools involved is not as easy as one would like. At this point, serious Grid deployment and use requires a group of knowledgeable, dedicated people. Hence, tools must be simpler for reliable deployment and use by nonspecialists.

Tools should also be configurable to the intended scope of deployment. Most Grid tools have been designed to be open-ended to support the concept of an open-ended “Grid” while, in fact, they have been used in an enterprise-scale deployment. While the ultimate vision is to have a ubiquitous service-oriented architecture, we must realize that a practical, evolutionary step is to have tools that can support the enterprise-scale Grid where many issues can be resolved, or “defined away,” by policy.

B. Standards—The Web/Grid Convergence

A key to defining exactly what the core Grid services are and facilitating their easy deployment on all scales is standards. To this end, the Global Grid Forum defined the Open Grid Services Architecture (OGSA) extending the Web Services (to support transient and stateful behaviors) and combined them with Grid protocols to define the Open Grid Services Infrastructure (OGSI), providing a uniform architecture for building and managing Grids and Grid applications. To permit the management of both stateful and stateless Web services, the Web Services Resource Framework (WSRF) was defined. This offers a potential alternative to OGSI and represents an opportunity for the proper convergence of Web and Grid service architectures. This convergence has tremendous importance, since it offers a solid platform for the further adoption of Web and Grid services in response to the significant economic motivations of the marketplace.

C. Nontechnical Barriers to Acceptance

Besides the technical issues concerning Grid adoption mentioned above, there are clearly many *nontechnical* or *cultural* barriers as well [13]. Grid computing, in many ways, is about *resource sharing*, while the “corporate culture” of many organizations may be fundamentally opposed to this. Some organizational units may jealously guard their machines or data out of a perceived economic or security threat. On a legal level, Grid computing may require the redefinition of ownership, copyrights, and licensing. Clearly, as Grid computing progresses, such cultural, legal, and economic issues will have to be resolved by adjusting our cultural policies and expectations to integrate what the technology will provide.

IV. OVERVIEW OF THE SPECIAL ISSUE

The overall goal of this special issue is to provide an overview of the current state in the field of Grid computing, including the state of the art of research in Grid applications, Grid model, environments, and tools, Grid architectures and infrastructures, and Grid computing trends and visions for the future. The papers in this issue include overviews of the field and its issues that target nonexperts, while including sufficient coverage and technical content to be a valuable reference for researchers in the field.

To this end, this special issue consists of four parts designed to lead the reader through a sequence of major topic areas. We start with a group of papers giving concrete descriptions of current Grids and Grid applications to illustrate what is possible and being done today to motivate the

broadest possible interest by specialists and nonspecialists alike. The next section of papers discusses tools and methods being used by Grid practitioners to build a variety of Grid applications. The third section explores the emerging, fundamental Grid architecture on which the global infrastructure will eventually depend. Finally, we present a section of papers focusing on the future direction and vision for Grid computing. We now summarize each part in turn.

Part I of this special issue focuses on Grid deployments and Grid applications, and includes papers describing representative deployments and applications in various disciplines of science and engineering. The first paper in this part, “The Earth System Grid: Supporting the Next Generation of Climate Modeling Research,” describes the Earth System Grid (ESG) that addresses the management, discovery, access, and analysis of very large, distributed datasets associated with the modeling and simulation of the earth’s climate. This project address core Grid computing issues (authentication, authorization, large-scale data transport and management, high-performance remote data access, scalable data replication, cataloging, data discovery, and distributed monitoring) in the context of this application. The next paper, “DAME: Searching Large Data Sets Within a Grid-Enabled Engineering Application,” describes the use of Grids in the aeroengine health-monitoring domain. This paper describes the Signal Data Explorer application developed within the DAME project, which uses advanced neural-network based methods to search for matching patterns in time-series vibration data originating from Rolls-Royce aeroengines. The large volume of data associated with the problem warrants the development of a distributed search engine, where data is held at a number of geographically disparate locations. The third paper, “The Computational Chemistry Prototyping Environment,” presents a Grid-based environment for computational chemistry consisting of a general scientific workflow environment, a domain-specific example for quantum chemistry, and the design of a workflow user interface, and efforts at database integration. The final paper in Part I, “Japanese Computational Grid Research Project: NAREGI,” describes one of the major Japanese national IT projects (National Research Grid Initiative) based on collaborations among industry, academia, and the government. The efforts consists of research and development in high-performance, scalable Grid middleware technologies, as well as research on leading-edge, Grid-enabled nanoscience and nanotechnology simulation applications.

Part II of the special issue focuses on models, environments, and tools for Grid computing. This part includes papers addressing programming paradigms, problem-solving environments, application development and management tools, and data-management and exploration systems. This first paper in this part, “The Grid Application Toolkit: Toward Generic and Easy Application Programming Interfaces for the Grid,” addresses the need for a high-level application programming toolkit, bridging the gap between existing Grid middleware and application-level needs. This paper describes the Grid Application Toolkit (GAT) that provides a unified programming interface to the Grid infrastructure tailored to the needs of the Grid application developer.

The software architecture of GridLab is also presented. The next paper, "Building Grid Portal Applications From a Web-Service Component Architecture," describes an application architecture based on wrapping user applications and application workflows as Web services and Web service resources. These services are visible through a family of Grid portal components, which can be used to configure, launch, and monitor complex applications built from the services. The third paper in this part, "Deploying the NaradaBrokering Substrate in Aiding Efficient Web and Grid Service Interactions," presents a messaging infrastructure for collaboration, peer-to-peer, and Grid applications. The paper also describes the integration of the NaradaBrokering substrate with Web Services. The final paper of Part II, "Data Grids, Digital Libraries, and Persistent Archives: An Integrated Approach to Sharing, Publishing, and Archiving Data," examines the synergies between data management systems, including Grids, Data Grids, digital libraries, and persistent archives and investigates the Grid infrastructure required for the generation, management, sharing, and preservation of information.

Part III of this issue concentrates on the Grid architecture, infrastructure, and middleware and includes research papers describing core efforts such as Legion, the WSRF, the OGSA, workflow management in Grid environments, resource discovery, resource management and scheduling, and security. This first paper in this part, "Legion: Lessons Learned Building a Grid Operating System," describes the Legion operating system-like Grid middleware, which provides a virtual machine interface layered over the Grid. The paper also describes the evolution of Legion and important lessons, both technical and sociological, learned in the process. The next paper, "Modeling and Managing State in Distributed Systems: The Role of OGSI and WSRF," introduces two approaches to modeling and manipulating state within a Web services framework: the OGSI and the WSRF. OGSI addresses the creation and management of a stateful Web service. WSRF refactors and evolves OGSI to exploit new Web services standards. The relationship between OGSI and WSRF is explained. The third paper in this part, "Coordination in Intelligent Grid Environments," explores the construction of intelligent computational Grids, where societal services exhibit intelligent behavior. The paper focuses on the coordination service that, acting as a proxy on behalf of end users, reacts to unforeseen events, plans how to carry out complex tasks, and learns from the history of the system. A prototype system used for a virtual laboratory in computational biology is presented. The fourth paper in the part, "Agreement-Based Resource Management," describes a unifying resource management framework, based on the concept of agreement-based resource management, to address the requirements of resource sharing in Grid environments. A general agreement model is presented and current resource management systems are examined in the context of this model. The final paper of Part III, "Security for Grids," addresses the security challenges of Grid environments. It characterizes security activities, examines the current state of the art, and introduces new

technologies that promise to meet the security requirements of Grids more completely.

Part IV outlines current trends, future directions, and visions for the Grid. The first paper in this part, "Conceptual and Implementation Models for the Grid," adopts models from distributed computing systems as a basis for defining and characterizing Grids and their programming models and systems. This paper motivates the need for a self-managing Grid computing paradigm and analyzes existing Grid programming systems that address this need. The second paper, "The Semantic Grid: Past, Present, and Future," presents the Semantic Grid as an extension of the current Grid in which information and services are given well-defined meaning. This paper outlines the requirements of the Semantic Grid, discusses the state of the art, and identifies the research challenges. The third paper, "Cyberinfrastructure for Science and Engineering: Promises and Challenges," describes the NSF's vision for a ubiquitous and accessible cyberinfrastructure that has the potential for revolutionizing all areas of science and engineering research and education. The paper also outlines some of the challenges and a possible path toward reaching the vision. The fourth paper in this part, "Grid Computing and Beyond: The Context of Dynamic Data Driven Applications Systems," introduces the Dynamic Data Driven Applications Systems (DDDAS) paradigm for the Grid, which is based on the ability to incorporate additional data into an executing application. The paper outlines the requirements of DDDAS and addresses the new capabilities and the technology challenges and opportunities of DDAS in Grid environments. The final paper of part IV, "The Grid Economy," proposes computational economy as a metaphor for effective management of resources and application scheduling in Grid environments. This paper also presents a service-oriented Grid architecture driven by Grid economy and commodity and auction models for resource allocation.

We would like to thank the authors for their excellent contributions to this special issue, and J. Calder, V. Prasanna, and the Editorial Board for their guidance and suggestions in putting the issue together. We also wish to thank C. Catlett for filling in several key historical aspects in the evolution of Grid computing.

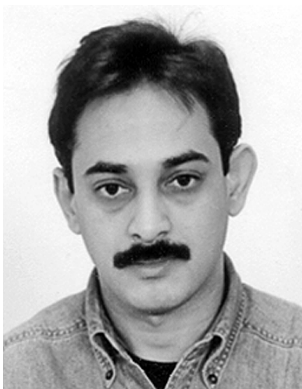
MANISH PARASHAR
Department of Electrical
and Computer Engineering
Rutgers University
Piscataway, NJ 08854 USA

CRAIG A. LEE
Computer Systems
Research Department
The Aerospace Corporation
El Segundo, CA 90245 USA

REFERENCES

- [1] Globus Alliance [Online]. Available: <http://www.globus.org>
- [2] Unicore Forum [Online]. Available: <http://www.unicore.org>

- [3] C. Catlett and L. Smarr, "Metacomputing," *Commun. ACM*, vol. 36, no. 6, pp. 44–52, 1992.
- [4] F. J. Corbat and V. A. Vyssotsky, "Introduction and overview of the multics system," in *Proc. Amer. Federation Information Processing Soc. 1965 Fall Joint Computer Conf.*, vol. 27, pp. 185–196.
- [5] T. DeFanti, I. Foster, M. Papka, R. Stevens, and T. Kuhfuss, "Overview of the I-WAY: Wide area visual supercomputing," *Int. J. Supercomput. Appl. High Perform. Comput.*, vol. 10, no. 2/3, pp. 123–131, 1996.
- [6] D. H. J. Epema, M. Livny, R. V. Dantzig, X. Evers, and J. Pruyne, "A worldwide flock of condors: Load sharing among workstation clusters," *J. Future Gener. Comput. Syst.*, vol. 12, pp. 53–65, 1996.
- [7] I. Foster, J. Geisler, W. Nickless, W. Smith, and S. Tuecke, "Software infrastructure for the I-WAY high performance distributed computing experiment," in *Proc. 5th IEEE Symp. High Performance Distributed Computing*, 1996, pp. 562–571.
- [8] I. Foster and C. Kesselman, Eds., *The Grid: Blueprint for a New Computing Infrastructure*. San Mateo, CA: Morgan Kaufmann, 1998.
- [9] I. Foster, C. Kesselman, and S. Tuecke, "The nexus task-parallel runtime system," in *Proc. 1st Int. Workshop Parallel Processing*, 1994, pp. 457–462.
- [10] A. S. Grimshaw and W. A. Wulf, "The legion vision of a worldwide virtual computer," *Commun. ACM*, vol. 40, no. 1, pp. 39–45, 1997.
- [11] H. Korab and M. Brown, Eds., (1995) *Virtual Environments and Distributed Computing at SC'95: GII Testbed and HPC Challenge Applications on the I-WAY*. [Online]. Available: <http://www.ncsa.uiuc.edu/General/Training/SC95/GII.HPCC.html>
- [12] C. Lee, C. Kesselman, and S. Schwab, "Near-real-time satellite image processing: Metacomputing in CC++," *IEEE Comput. Graph. Appl.*, vol. 16, no. 4, pp. 79–84, Jul. 1996.
- [13] The politics of Grid: Organizational politics as a barrier to implementing Grid computing (2004). [Online]. Available: <http://www.platform.com/adoption/politics>
- [14] D. Skillicorn and D. Talia, "Models and languages for parallel computation," *ACM Comput. Surv.*, vol. 30, no. 2, pp. 123–169, 1998.
- [15] R. Stevens, P. Woodward, T. DeFanti, and C. Catlett, "From the I-WAY to the national technology grid," *Commun. ACM*, vol. 40, no. 11, pp. 51–60, 1997.
- [16] W. Schatz, "Computing without boundaries," *Inf. Week*, August 3, 1992.



Manish Parashar (Senior Member, IEEE) received the B.E. degree in electronics and telecommunications from Bombay University, Bombay, India in 1988 and the M.S. and Ph.D. degrees in computer engineering from Syracuse University, Syracuse, NY, in 1994

He is Associate Professor of Electrical and Computer Engineering at Rutgers University, Piscataway, NJ, where he also is director of the Applied Software Systems Laboratory. He has coauthored over 130 technical papers in international journals and conferences, has coauthored/edited five books/proceedings, and has contributed to several others in the area of parallel and distributed computing. His current research interests include autonomic computing, parallel, distributed, and Grid computing, networking, scientific computing, and software engineering.

Dr. Parashar is a Member of the Association for Computing Machinery. He has received the National Science Foundation's CAREER Award (1999) and the Enrico Fermi Scholarship from the Argonne National Laboratory (1996). He is a Member of the Executive Committee

of the IEEE Computer Society Technical Committee on Parallel Processing (TCPP), part of the IEEE Computer Society Distinguished Visitor Program (2004–2006). He is also the Cofounder of the IEEE International Conference on Autonomic Computing (ICAC).



Craig A. Lee (Member, IEEE) received the B.A. degree in psychology from Reed College, Portland, OR, in 1975, the M.S. degree in computer science from Syracuse University, Syracuse, NY, in 1982, and the Ph.D. degree in computer science from the University of California, Irvine, in 1988.

He is a Section Manager in the Computer Systems Research Department, The Aerospace Corporation, El Segundo, CA, a nonprofit, federally funded, research and development center. He has worked in the area of parallel and distributed computing for the last 25 years. He has built many application prototypes with a strong focus on experimental languages, tools, and environments. He has also conducted Defense Advanced Research Projects Agency- and National Science Foundation (NSF)-sponsored research in the areas of Grid computing, optimistic models of computation, active networks, and distributed simulations, in collaboration with the University of Southern California, Los Angeles; the University of California, Los Angeles (UCLA); California Institute of Technology, Pasadena; the Argonne National Laboratory, Argonne, IL; and the College of William and Mary, Williamsburg, VA. He has coauthored over 50 technical works, including four book chapters and seven edited volumes and issues and has recently joined the Editorial Board of *Future Generation Computing Systems*.

Dr. Lee is a Member of the Association for Computing Machinery and the IEEE Computer Society, and has lectured undergraduate computer science courses at UCLA. He has served as Area Cochair of the Applications, Programming Models, and Environments Area of the Global Grid Forum and as Cochair of the GridRPC Working Group. He is also on the Steering Committees of Cluster Computing and the Grid (CCGrid) and the International Workshop on Grid Computing. He has served on the Program Committee for many other conferences and workshops and has served as a Panelist for the NSF and NASA and as an External Evaluator for the Institut National de Recherche en Informatique et en Automatique (INRIA).

Dr. Lee is a Member of the Association for Computing Machinery and the IEEE Computer Society, and has lectured undergraduate computer science courses at UCLA. He has served as Area Cochair of the Applications, Programming Models, and Environments Area of the Global Grid Forum and as Cochair of the GridRPC Working Group. He is also on the Steering Committees of Cluster Computing and the Grid (CCGrid) and the International Workshop on Grid Computing. He has served on the Program Committee for many other conferences and workshops and has served as a Panelist for the NSF and NASA and as an External Evaluator for the Institut National de Recherche en Informatique et en Automatique (INRIA).