Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
J. Parallel Distrib. Comput. 64 (2004) 689–691
Journal of Parallel and Distributed Computing
http://www.elsevier.com/locate/jpdc

Preface

Middleware: the key to next generation computing

According to Backus [1], the computing revolution has experienced three waves. The first wave began with commercialization of the silicon chips, and lasted 10–15 years. The second wave was about software and began sometime in the mid-1980s. The third wave began in the late 1990s and was about networking and communication. The third wave is credited for the last IT boom. Growth in technology has made computing devices ever smaller and faster then before. However, after the dot com bubble, where is the breakthrough of the next IT boom? Several new disciplines of computing have emerged during the last few years. Every indication shows that the third wave of computing revolution is still not over. The new developments still involve network computing, and middleware is the key technology.

Fig. 1 illustrates the evolution of network-based computing relating the two new directions of computing: grid computing and pervasive computing. It is an extended version of M. Satyanarayan's vision of pervasive computing [3]. Grid computing coordinates network resources and mimics an electrical power grid by bringing remote computing power uniformly and transparently to the users [2]. Pervasive computing emphasizes everywhere, all-time services and a "human-centered" view of computing. It forms the "smart space" in which IT service is as naturally available as the air we breathe. While distributed computing and mobile computing are often not considered new technologies, a number of technical issues have not been resolved and are facing new complexities. For instance, security is much more difficult to implement in distributed systems than in separate, independent systems.

Satyanarayanan [3] identifies four new research thrusts that need to be addressed to make pervasive computing a reality: effective use of smart space, invisibility, localized scalability, and masking uneven conditions. Invisibility here means minimal user distraction. Scalability is in the normal sense of software scalability. Pervasive computing demands more interaction between user personal space and its surroundings through smart entities located in the user environment. In this scenario, newer systems should provide better scalability. The term "localized" indicates that scalability is within each smart space. Masking uneven conditions is the challenge of developing a smart space under different infrastructures, which include non-technical factors such as organizational structure, economics, and business models. The most challenging thrust probably is the effective use of smart space. Here "space" is a physical area covered by coordinated embedded systems. "Smart" indicates that the coordinated embedded system is equipped with fusion sensors that can monitor/capture human activities and with context-aware middleware that can understand human intentions. "Effective use" means that the smart space can provide satisfactory "human-centered" service and meet the expectations of users.

Similar to pervasive computing, grid computing is also driven by four thrust patterns as shown in Fig. 1: effective use of federated communities, virtualization, standardization, and masking of uneven conditions. A community, or a virtual organization, is a set of distributed network resources managed under the same rules or the same administration domain. Federation means these communities are coordinated for resources sharing and problem solving. Effective use indicates the use of the federated resources to deliver nontrivial qualities of services. Virtualization means the resources in the communities are virtualized to hide its ownership and location. Standardization implies that the federation is built on standard, open, general-purpose protocols and interfaces. The "masking of uneven conditions" in grid computing is different from that in pervasive computing. In pervasive computing, the smart space middleware system has to be developed to match the infrastructure of the underlying physical space. In grid computing, the federated middleware system has to blend in with the middleware of the underlying virtual organizations. While the key to pervasive computing is middleware technology, the grid environment itself is a middleware designed to coordinate other middleware.

Middleware refers to the software that is common to multiple applications and builds on the network transport services to enable ready development of new applications and network services. It provides an abstract interface that gives an application developer a uniform view of low-level operating systems and networks. More precisely, middleware allows scientists and engineers to transparently use and share distributed

0743-7315/$ - see front matter
doi:10.1016/j.jpdc.2004.03.002

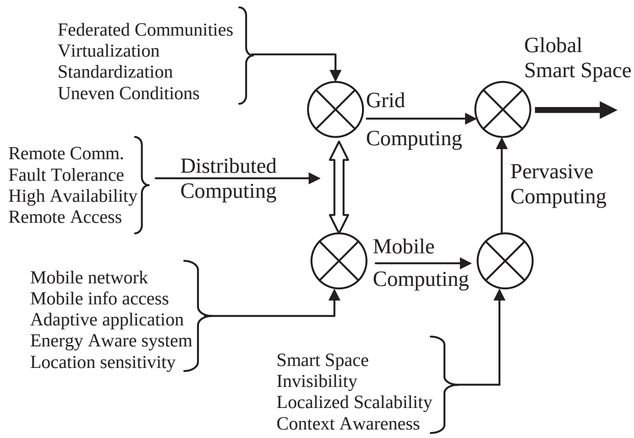

Fig. 1. The evolution of network-based computing.

resources such as computers, data, networks, and instruments; supports effective collaboration and communications tools to expedite research and education; and provides working architectures and approaches that can be extended to the larger set of Internet and network users. Middleware makes things transparent to the end user, providing consistency, security, privacy, and capabilities. Middleware is essential for Internet computing, distributed computing, and high-performance parallel computing, attracting successful research from both academia and industry.

The nine articles in this special issue represent leading-edge middleware research. They are selected from three different research focuses: grid technology, middleware for pervasive computing, and application support. In grid technology, the focus is on resource management. In middleware support for pervasive computing, the focus is on context awareness and capacity awareness. The selection in application support is diverse, one for system support, one for language support, and one for application support.

As shown in Fig. 1, pervasive computing is mainly the merger of mobile computing and smart space. The merger requires the mobile devices to be dynamically integrated with the smart space, or the local network computing environment in general, in which they have moved into and have conducted pervasive service. Yau and Karim have developed a context-sensitive middleware, called re-configurable context-sensitive middleware (RCSM), for dynamic integration of mobile devices with network infrastructures (NI). In a sense, RCSM is a leading middleware in this merge, as it allows both mobile devices and NI to interoperate with each other; this interoperation can be restricted on specific contexts and provides transparency over the dynamic resource discovery and networking. The mobile device may have to interoperate with the NI to receive the necessary service or may just like to offload the work to get a better performance or save energy. Based on a compiler approach, Wang and Li present an offloading scheme to find an optimal offloading partition for improving performance and energy consumption that guarantee its correctness. Distributed computing traditionally follows the client-server model. A client sends requests. A server provides service. Similarly, grid computing is also based on the client-server model. In a pervasive environment, all the computing entities may be equal. They are peers, sending requests and providing services. Different from client-server model, peer-to-peer technology has drawn intensive attention recently. Its most successful application is in distributed file systems where each peer is treated as a first class entity in the system and can store and retrieval data. Hsiao and King present their experience in developing Tornado, a peer-to-peer storage network. A unique feature of Tornado is its ability to be aware of the capabilities of the underlying systems to cope with system heterogeneity and to leverage the performance and reliability of the storage network.

Grid computing is all about coordination and management. Three articles on managerial middleware are selected. Salceda et al. provide general middleware architecture for distributed system management. They use an object-oriented model to represent management information. All management information, then, is stored without redundancies into a common, network-accessible repository, where any managed element can query, update, or retrieve the information stored through a common interface. The Condor group at Wisconsin–Madison addresses the core issue of grid management, efficient resource management. They consider both application performance optimization and system utilization and propose a dynamic scheduling strategy for master–worker applications in a shared environment. The dynamic scheduling strategy is based on a self-adjusting approach, which automatically adjusts resources used by the grid application in a shared environment. The adjustment is designed to optimize a balanced system utilization and application speedup based on local resources usage. Prodan and Fahringer's work is on a different management. Their middleware, named ZENTURIO, is a toolkit developed to manage experimental testing in a grid and distributed computing environment. Software development involves many cycles of code editing, execution, testing, performance analysis, and tuning. It may require repeated execution of the same application with various parameters for performance optimization and correctness verification. ZENTURIO is designed based on grid and Web service technology and for supporting OGSA-compliant grid service development. It is a grid service for grid application development.

Middleware is software used by a large class of distributed applications. One of the main challenges of distributed environment is to achieve scalability of synchronization. Desai and Mueller provide a middleware protocol to support scalable synchronization and state replication in distributed systems. The protocol is a hierarchical locking protocol with a peer-to-peer paradigm, in the sense it relies on fully decentralized data structures and symmetric algorithms to ensure scalability. The combination of hierarchical structure and peer-to-peer coordination makes their approach highly scalable for transaction-style processing and distributed agreement. Desai and Mueller implement their protocol under CORBA with modified specifications. CORBA is one of the most popular commercial middleware standards for distributed object computing [4]. It is inadequate, however, in describing the overall software architecture and dynamically reconfiguring the set of software components according to the requirements of the applications and their operational environments. Cao et al. describe a novel framework for architecting and high-level programming support of CORBA distributed applications. It provides a reflective, architectural approach to structuring and reconfiguring CORBA-based distributed applications. Vadhiyar and Dongarra focus on middleware support for computational intensive applications. A Web service, or a network server, provides service to the clients. For computational intensive applications, the server may in turn invoke a cluster of network resources to conduct parallel computing. Traditionally, the server's parallel processing is disconnected with the client application task scheduling, which makes global performance and utilization optimization hard to achieve. Vadhiyar and Dongarra develop a grid-based system that allows users to invoke MPI applications on remote grid resources from a sequential environment and selects the appropriate network resources for server processing based on the resources availability and application characteristics.

To summarize, middleware is the key to next generation computing. From grid computing to pervasive computing and from Web service to peer-to-peer technology, the major gluing element is middleware technology. It is for this reason that there has been significant effort at the national level for middleware development and support. To leverage and advance the potential of the middleware technology, the National Science Foundation Middleware Initiative (NMI) program was initiated in 2001 [5]. Several papers included in this special issue are supported by NMI. Middleware research has already yielded strong empirical and quantitative results. The papers selected are only the tip of the iceberg. Middleware will seamlessly integrate application computation with network infrastructure and will become the foundation for the next IT revolution that will usher in a new wave of technology to benefit both research and business.

References

[1] J. Backus, Funding the computing revolution's third wave, Commun. ACM 44(11) (2001) 71–76.
[2] I. Foster, C. Kesselman, The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, Los Altos, CA, 2004.
[3] M. Satyanarayanan, Pervasive computing: vision and challenges, IEEE Personal Commun. (2001) 10–17.
[4] Object Management Group, The Common Object Request Broker: Architecture and Specification, Version 3.0, 2002. Available at http://www.omg.org/.
[5] The National Science Foundation Middleware Initiative (NMI), http://www.nsf-middleware.org/.

Xian-He Sun
Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616, USA
E-mail address: sun@iit.edu

Alan R. Blatecky
San Diego Supercomputing Center, University of California, San Diego, La Jolla, CA 92093-0505, USA
E-mail address: alan@sdsc.edu