1

MARKET-ORIENTED COMPUTING AND GLOBAL GRIDS: AN INTRODUCTION

Rajkumar Buyya and Srikumar Venugopal

1.1 INSPIRATION

Following Alessandro Volta's invention of the electric battery in 1800, Thomas Edison and Nikola Tesla paved the way for electricity's widespread use by inventing the electric bulb and alternating current (AC), respectively. Figure 1.1 shows Volta demonstrating the battery for Napoleon I in 1801 at the French National Institute, Paris. Regardless of whether Volta envisioned it, his invention evolved into a worldwide electrical power Grid that provides dependable, consistent, and pervasive access to utility power and has become an integral part of modern society [1].

Inspired by the electrical power Grid's pervasiveness, ease of use, and reliability, computer scientists in the mid-1990s began exploring the design and development of an analogous infrastructure, called the *computational power Grid* [4], for wide-area parallel and distributed computing [6]. The motivation for computational Grids was initially driven by large-scale, resource (computational and data)-intensive scientific applications that require more resources than a single computer [PC, workstation (WS), supercomputer, or cluster] could provide in a single administrative domain. A Grid enables the sharing, selection, and aggregation of a wide variety of geographically distributed resources, including supercomputers, storage systems, data sources, and specialized devices owned by different organizations for solving large-scale resource-intensive problems in science, engineering, and commerce. Because of its potential to make impact on the twenty-first century as much as the electric power

Market-Oriented Grid and Utility Computing Edited by Rajkumar Buyya and Kris Bubendorfer

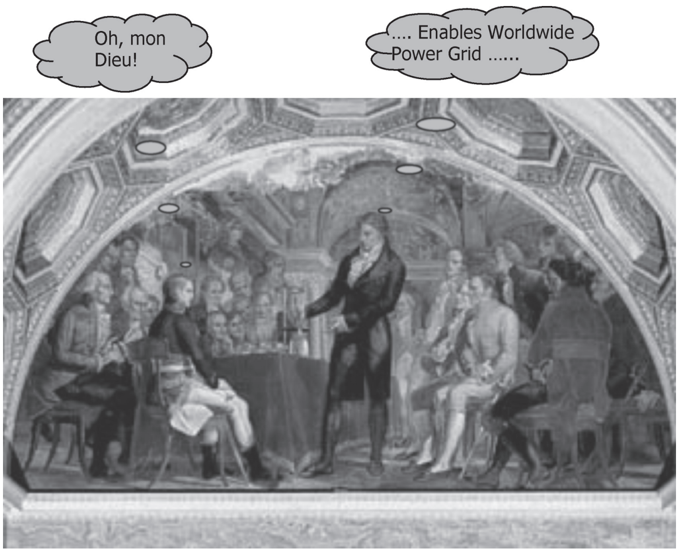

Figure 1.1 Volta demonstrates the battery for Napoleon I at the French National Institute, Paris, in 1801. The painting (by N. Cianfanelli, 1841) is from the Zoological Section of "La Specula" at the National History Museum, Florence University, Italy.

Grid did on twentieth century, Grid computing has been hailed as the next revolution after the Internet and the Web.

These developments foreshadow the realization of the vision of Leonard Kleinrock, one of the chief scientists of the original Advanced Research Projects Agency Network (ARPANET) project that seeded the Internet, who said in 1969 [3]: "As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of '*computer utilities,*' which, like present electric and telephone utilities, will service individual homes and offices across the country."

Utility computing is envisioned to be the next generation of information technology (IT) evolution that depicts how computing needs of users can be fulfilled in the future IT industry [13]. Its analogy is derived from the real world, where service providers maintain and supply utility services, such as electrical power, gas, and water to consumers. Consumers in turn pay service providers according to their usage. Therefore, the underlying design of utility computing is based on a service provisioning model, where users (consumers) pay providers for using computing power only when they need to.

1.2 GRID COMPUTING

Grid computing follows the service-oriented architecture and provides the hardware and software services and infrastructure for secure and uniform access to heterogeneous resources and enables the formation and management of virtual organizations (VOs). It also supports application and services composition, workflow expression, scheduling, and execution management and service-level agreement (SLA)-based allocation of resources.

As there are a large number of projects around the world working on developing Grids for different purposes at different scales, several definitions of Grid abound. The Globus project (Argonne National Laboratory, USA) defines Grid as “an infrastructure that enables the integrated, collaborative use of high-end computers, networks, databases, and scientific instruments owned and managed by multiple organizations.” Another utility notion-based Grid definition put forward by the Gridbus project (University of Melbourne, Australia) is “Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed ‘autonomous’ resources dynamically at runtime depending on their availability, capability, performance, cost, and users’ Quality of Service (QoS) requirements.”

The development of the Grid infrastructure, both hardware and software, has become the focus of a large community of researchers and developers in both academia and industry. The major problems being addressed by Grid developments are the social problems involved in collaborative research:

- Improving distributed management while retaining full control over locally managed resources
- Improving the availability of data and identifying problems and solutions to data access patterns
- Providing researchers with a uniform user-friendly environment that enables access to a wider range of physically distributed facilities improving productivity

A high-level view of activities involved within a seamless and scalable Grid environment is shown in Figure 1.2. Grid resources are registered within one or more

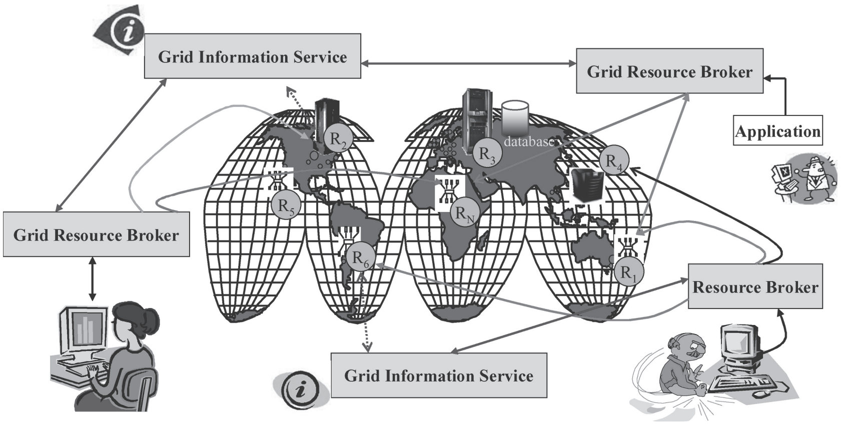

Figure 1.2 A worldwide Grid computing environment.

Grid information services. The end users submit their application requirements to the Grid resource broker, which then discovers suitable resources by querying the information services, schedules the application jobs for execution on these resources, and then monitors their processing until they are completed. A more complex scenario would involve more requirements and therefore, Grid environments involve services such as security, information, directory, resource allocation, application development, execution management, resource aggregation, and scheduling. Software tools and services providing these capabilities to link computing capability and data sources in order to support distributed analysis and collaboration are collectively known as *Grid middleware*.

In order to provide users with a seamless computing environment, the Grid middleware systems need to solve several challenges originating from the inherent features of the Grid [8]. One of the main challenges is the heterogeneity that results from the vast range of technologies, both hardware and software, encompassed by the Grid. Another challenge involves the handling of Grid resources that are spread across political and geographic boundaries and are under the administrative control of different organizations. It follows that the availability and performance of Grid resources are unpredictable as requests from within an administrative domain may gain higher priority over requests from outside. Thus, the dynamic nature of Grid environment poses yet another challenge.

To tackle these challenges, a Grid architecture has been proposed based on the creation of virtual organizations (VOs) [9] by different physical (real-world) organizations coming together to share resources and collaborating in order to achieve a common goal. A VO defines the resources available for the participants and the rules for accessing and using the resources. Within a VO, participants belonging to member organizations are allocated resource shares according to the urgency and priority of a request as determined by the objectives of the VO. Another complementary Grid architecture [10] is based on economic principles in which resource providers (owners) compete to provide the best service to resource consumers (users) who select appropriate resources according to their specific requirements, the price of the resources, and their quality-of-service (QoS) expectations from the providers. Two examples of QoS terms are the deadline by which the resource needs to be available and the maximum price (budget) that can be paid by the user for the service. QoS terms are enforced via service-level agreements (SLAs) between the providers and the consumers, the violation of which results in penalties.

1.3 GRID COMPONENTS

In a worldwide Grid environment, capabilities that the infrastructure needs to support include

- Remote storage and/or replication of datasets
- Publication of datasets using global logical name and attributes in the catalog
- Security—access authorization and uniform authentication

- Uniform access to remote resources (data and computational resources)
- Publication of services and access cost
- Composition of distributed applications using diverse software components including legacy programs
- Discovery of suitable datasets by their global logical names or attributes
- Discovery of suitable computational resources
- Mapping and scheduling of jobs (aggregation of distributed services)
- Submission, monitoring, and steering of job execution
- Movement of code/data between user desktop computers and distributed Grid resources
- Enforcement of QoS requirements
- Metering and accounting of resource usage

These capabilities in Grid computing environments play a significant role in enabling a variety of scientific, engineering, and business applications. Various Grid components providing these capabilities are arranged into layers. Each layer builds on the services offered by the lower layer in addition to interacting and cooperating with components at the same level (e.g., resource broker invoking secure process management services provided by core middleware). Figure 1.3 shows four layers of the hardware and software stack within a typical Grid architecture: fabric, core middleware, user-level middleware, and applications/portals layers. Adaptive management capabilities are supported by implementing principles of market-oriented resource management mechanisms in different horizontal layers.

The *Grid fabric* layer consists of distributed resources such as computers, networks, storage devices, and scientific instruments. The computational resources

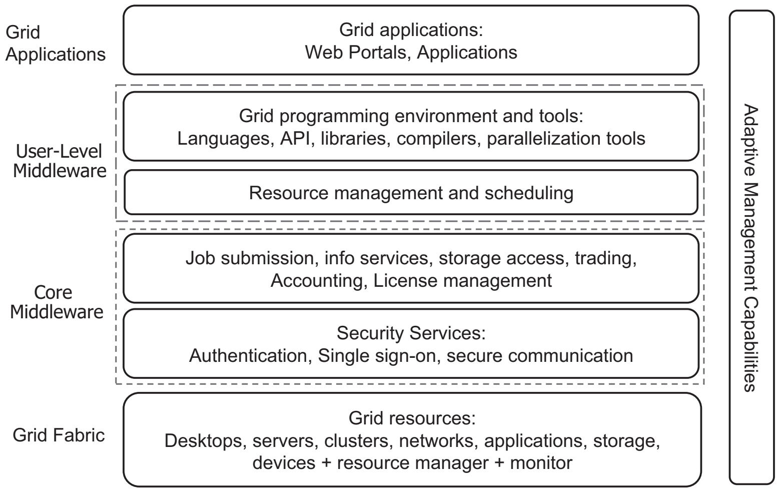
Grid Applications
Grid applications: Web Portals, Applications
User-Level Middleware
Grid programming environment and tools: Languages, API, libraries, compilers, parallelization tools
Resource management and scheduling
Core Middleware
Job submission, info services, storage access, trading, Accounting, License management
Security Services: Authentication, Single sign-on, secure communication
Grid Fabric
Grid resources: Desktops, servers, clusters, networks, applications, storage, devices + resource manager + monitor
Adaptive Management Capabilities

Figure 1.3 A layered Grid architecture and components.

represent multiple architectures such as clusters, supercomputers, servers, and ordinary PCs that run a variety of operating systems (such as UNIX variants or Windows). Scientific instruments such as telescopes and sensor networks provide real-time data that can be transmitted directly to computational sites or are stored in a database.

The *core Grid* middleware offers services such as remote process management, coallocation of resources, storage access, information registration and discovery, security, and aspects of QoS such as resource reservation and trading. These services abstract the complexity and heterogeneity of the fabric level by providing a consistent method for accessing distributed resources.

The *user-level Grid* middleware utilizes the interfaces provided by the low-level middleware to provide higher-level abstractions and services. These include application development environments, programming tools, and resource brokers for managing resources and scheduling application tasks for execution on global resources.

Grid applications and portals are typically developed using Grid-enabled programming environments and interfaces and are deployed on Grids using brokering and scheduling services provided by user-level middleware. An example application, such as parameter simulation of a grand-challenge problem, would require computational power and access to remote datasets, and may need to interact with scientific instruments. Grid portals offer Web-enabled application services, where users can submit their jobs to remote resources and collect results from them through the Web.

The design aims and benefits of Grids are analogous to those of utility computing, thus highlighting the potential and suitability of Grids to be used as utility computing environments. The current trend of implementing Grids based on open standard service-based architectures to improve interoperability is a step toward supporting utility computing. Even though most existing Grid applications are scientific research and collaboration projects, the number of applications in business and industry-related projects is gradually increasing. It is thus envisioned that the realization of utility computing through Grids will follow a course similar to that of the World Wide Web, which was first initiated as a scientific project but was later widely adopted by businesses and industries.

1.4 GRID INITIATIVES AROUND THE WORLD

Given the possibilities of Grid computing, it is no surprise that there is keen interest in this technology around the world (globally). Currently, Grid projects that have been initiated on the global scale can be broadly classified into two categories [8]: (1) Grid infrastructure development, which involves setting up hardware, software, and administrative mechanisms to enable application scientists to make use of these facilities for their research; and (2) Grid middleware research, which investigates the development of software and policy mechanisms that assist in realizing the full potential of Grid computing. Many of these projects are motivated by large-scale scientific projects that will involve the production and analysis of data at an unprecedented scale. One frequently cited such large-scale scientific project is the Large Hadron Collider (LHC) experiments [11] at the European Organisation for Nuclear Research (CERN), which began data production in 2008. The volume of data generated by these experiments is in the petabyte (PB) range, for distribution to physicists around the world for analysis. As the Grid has been mandated as the IT infrastructure for handling the massive workloads of LHC experiments, all the collaborating nations are setting up Grid infrastructure in one form or another. In the following sections, we will describe some of the major Grid infrastructure and middleware projects around the world.

1.4.1 United States of Amercia (USA)

Production Grid testbeds for various application domains have been deployed over physical (hardware) Grid infrastructure such as the National Science Foundation (NSF)-funded TeraGrid [17] in the United States, which provides over 40 tera-floating-point operations per second (Tflops) of computing power at eight sites around the country with 2 PB of available storage interconnected by a network operating at a speed of 10–30 gigabits per second (Gbps). The BioInformatics Research Network (BIRN) is another testbed for the purpose of furthering biomedical science by sharing data stored in different repositories around USA. The NEESGrid enables scientists in the earthquake engineering community to carry out experiments in distributed locations and analyse data through a uniform interface.

Of the Grid middleware efforts in the United States, the Globus toolkit from the Globus Alliance led by Argonne National Laboratory is the most widely known. Other notable efforts are the Condor project (University of Wisconsin, Madison) for high-throughput computing mechanisms, and “i Rule Oriented Data Systems” (iRODS) [58] from the San Diego Supercomputing Center (SDSC) for Grid data management. In addition, several commercial organizations such as IBM, Sun Microsystems, Hewlett-Packard (HP), and Oracle are actively involved in the development of enterprise and global utility Grid technologies.

1.4.2 Europe

Two pioneering Grid efforts in Europe, started in early 2001, were the United Kingdom (UK)’s e-Science program [5] and the European Union (EU)-funded Data Grid project [11]. The latter was succeeded by the EGEE (Enabling Grids for E-sciencE) project, which aims to create a Grid infrastructure available to scientists and to develop robust middleware for application deployment. CERN manages the LHC Computing Grid (LCG) project, which has created a production Grid infrastructure for researchers involved in the experiments using the LHC.

Other notable EU-funded projects include GridLab [53], providing a Grid application development toolkit; Cactus framework, for scientific programming; GridSphere, for creating a Web portal environment for Grid users; P-Grade, providing a visual environment for application development; Triana, for workflow formulation; and OGSA-DAI, for integration of relational databases in Grid environments.

1.4.3 Asia–Pacific

Several countries in the Asia–Pacific region have started national Grid programs similar to those initiated in the United States and Europe. In addition, countries such as Australia, China, Japan, South Korea, and Singapore are active participants in worldwide Grid projects such as the LCG. Some of the notable Grid programs are the National Research Grid Initiative (NAREGI) in Japan, China National Grid in China, K*Grid in South Korea, and Garuda National Grid in India.

Prominent Grid middleware projects include the Ninf project (Tokyo Institute of Technology) for building a Grid-based remote procedure call (RPC) system [29], the Grid Datafarm (Gfarm) project (AIST, Japan) for providing a petascale data storage–processing system, the Nimrod/G project (Monash University, Australia) for parametric computations on Grid resources [28], and the Gridbus project (University of Melbourne, Australia) for market-oriented Grid and utility computing [54].

1.4.4 Standardization Efforts

Given the large amount of middleware development happening in this area of research, standardization is important to ensure interoperability between different products and implementations. Grid standardization efforts led by the Open Grid Forum (OGF) [12] have produced standards for almost all aspects of Grid technology. Work at the OGF has produced the open Grid service infrastructure (OGSI) specification and its successor, the Web services resource framework (WSRF), which have paved the way for integration of Web services within Grid architecture. This is important as Web services allow Grid developers to take advantage of standard message formats and mechanisms such as HTTP and XML for communicating between heterogeneous components and architectures. Other standardization bodies such as World Wide Web Consortium (W3C), Organization for Advancement of Structured Information Standards (OASIS), and Internet Engineering Task Force (IETF) also produce standards relevant to aspects of Grid Computing.

1.5 MARKET-ORIENTED GRID RESOURCE MANAGEMENT

Resource management and scheduling in Grid environments is a complex undertaking. The geographic distribution of resources owned by different organizations with different usage policies, cost models, and varying load and availability patterns is problematic. The *producers* (resource owners) and *consumers* (resource users) have different goals, objectives, strategies, and requirements. Classical Grids are motivated by an assumption that coordinated access to diverse and geographically distributed resources is valuable. However, this paradigm needs mechanisms that allow not only such coordinated access but also sustainable, scalable models and policies that promote utility-oriented sharing of Grid resources.

To address these resource management challenges, several groups of researchers have proposed a distributed computational economy-based[1] framework [10,14,50,56], for resource allocation and to regulate supply and demand of the available resources. This economy-based framework offers an incentive to resource owners for contributing and sharing resources, and motivates resource users to think about tradeoffs between the processing time (e.g., deadline) and computational cost (e.g., budget), depending on their QoS requirements. It can be observed that, even in electricity Grids, bid-based electricity trading over the Internet has been adopted to develop competitive forces in the electricity marketplace [20].

Resource management and scheduling systems for Grid computing need to manage resources and application execution depending on resource consumers' and owners' requirements, and they need to continuously adapt to changes in the availability of resources. This requirement introduces a number of challenging issues that need to be addressed, namely: site autonomy, heterogeneous substrate, policy extensibility, resource allocation or coallocation, online control, resource trading, and QoS-based scheduling.

1.5.1 Assessing Wants and Needs

In an economy-based Grid computing environment, resource management systems need to provide mechanisms and tools that allow resource consumers (end users) and providers (resource owners) to express their requirements and facilitate the realization of their goals. Resource consumers need

- A utility model to determine how consumers demand resources and their preference parameters
- A broker that supports resource discovery and strategies for application scheduling on distributed resources dynamically at runtime depending on their availability, capability, and cost along with user-defined QoS requirements

Resource providers need

- Tools and mechanisms that support price specification and generation schemes to increase system utilization
- Protocols that support service publication, trading, and accounting

For the market to be competitive and healthy, coordination mechanisms are required to help reach equilibrium price—the market price at which the supply of a service equals the quantity demanded.

1.5.2 Computational Economy and Its Benefits

Like all systems involving goals, resources, and actions, computations can be viewed in economic terms. With the proliferation of networks, high-end computing systems

[1] The terms "economic/economy-based" and "market-based" are synonymous and interchangeable.

architecture has moved from centralized toward decentralized models of control and action; the use of economy-driven market mechanisms would be a natural extension of this development. The ability of trade and price mechanisms to combine local decisions by diverse entities into globally effective characteristics reflect their value for organizing computations in large systems such as Internet-scale computational Grids.

The need for an economy-driven resource management and scheduling system comes from the answers to the following questions:

- What constitutes the Grid, and who owns its resources?
- What motivates resource owners to contribute their resources to the Grid?
- Is it possible to have access to all resources in the Grid by contributing our resource?
- If not, how do we have access to all Grid resources?
- If we have access to resources through collaboration, are we allowed to use them for any other purposes?
- Do resource owners charge the same or a different price for different users?
- Is access cost the same for peak and off-peak hours?
- How can resource owners maximize their profits?
- How can users solve their problems within a minimum cost?
- How can a user get high priority over others?
- If the user relaxes the deadline by which results are required, can solution cost be reduced?

Several individuals or organizations that have contributed resources to the Grid have been motivated largely by the public good, prizes, fun, fame, or collaborative advantage. This is clearly evident from the construction of private Grids (but on volunteer resources) or research testbeds such as SETI@Home [18], Condor pool [38], and TeraGrid [17]. The chances of gaining access to such computational testbeds for solving commercial problems are low. Furthermore, contributing resources to a testbed does not guarantee access to all the other resources in that testbed.

Commercial companies such as Entropia, ProcessTree, Popular Power, United Devices, and DataSynapse are exploiting idle central processing unit (CPU) cycles from desktop machines to build a commercial computational Grid infrastructure based on peer-to-peer (P2P) networks [19]. These companies are able to develop large-scale infrastructure for Internet computing and use it for their own financial gain by charging for access to CPU cycles for their customers, without offering fiscal incentive to all resource contributors. However, in the long run, this model does not support the creation of a maintainable and sustainable infrastructure, as the resource contributors have no incentive for their continued contribution. Therefore, a Grid economy seems a better model for managing and handling requirements of both Grid providers and consumers. The benefits of economy-based resource management include the following:

- It helps in building a large-scale Grid as it offers incentive for resource owners to contribute their (idle) resources for others to use and profit from.
- It helps in regulating the supply and demand for resources.
- It offers an economic incentive for users to back off when solving low-priority problems and thus encourages the solution of time-critical problems first.
- It removes the need for a central coordinator (during negotiation).
- It offers uniform treatment of all resources; that is, it allows trading of everything including computational power, memory, storage, network bandwidth/latency [22], data, and devices or instruments.
- It allows users to express their requirements and objectives.
- It helps in developing scheduling policies that are user-centric rather than system-centric.
- It offers an efficient mechanism for allocation and management of resources.
- It helps in building a highly scalable system as the decisionmaking process is distributed across all users and resource owners.
- It supports a simple and effective basis for offering differentiated services for different applications at different times.
- Finally, it places the power in the hands of both resource owners and users, enabling them to make their own decisions to maximize the utility and profit.

1.6 REQUIREMENTS FOR ECONOMY-BASED GRID SYSTEMS

To deliver value to users greater than that possible with traditional systems, economy-based resource management systems need to provide mechanisms and tools that allow resource consumers (end users) and providers (resource owners) to express their requirements and facilitate the realization of their goals. In other words, they need (1) the means to express their requirements, valuations, and objectives (*value expression*); (2) scheduling policies to translate them into resource allocations (*value translation*); and (3) mechanisms to enforce selection and allocation of differential services, and dynamic adaptation to changes in their availability at runtime (*value enforcement*). Similar requirements are raised [2] for market-based systems in a single-administrative-domain environment such as clusters, and they are limited to cooperative economic models since they aim for social welfare. Grids need to use *competitive economic models* as different resource providers and resource consumers have different goals, objectives, strategies, and requirements that vary with time.

Essentially, resource consumers need a *utility model* to allow them to specify resource requirements and constraints. For example, the Nimrod/G broker allows users to specify the deadline and budget constraints along with optimization parameters such as optimizing for time (*value expression*). They need *brokers* that provide strategies for choosing appropriate resources (*value translation*) and dynamically adapt to changes in resource availability at runtime to meet user requirements (*value enforcement*). The resource owners need mechanisms for *price generation schemes* to increase system utilization and *protocols* that help them offer competitive services (*value expression*). For the market to be competitive and healthy, coordination mechanisms are required that help the market reach an equilibrium price—the price at which the supply of a service equals the quantity demanded. Grid resources have their schedulers (e.g., OS or queuing system) that allocate resources (*value translation*). A number of research systems have explored QoS-based resource (e.g., CPU time and network bandwidth [22,23]) allocation in operating systems and queuing systems, but the inclusion of QoS into mainstream systems has been slow-paced (e.g., the Internet mostly uses the best effort allocation policy [24], but this is changing with IPv6 [25]). Some research systems support resource reservation in advance (e.g., reserving a slot from time t_1 to t_2 using the Globus Architecture for Reservation and Allocation (GARA) [21] and binding a job to it) and allocate resources during reserved time (*value enforcement*).

1.7 MARKET-ORIENTED GRID ARCHITECTURE

A reference service-oriented architecture for market-oriented Grids is shown in Figure 1.4. The key players in a market-oriented Grid are the Grid user, Grid resource broker, Grid middleware services, and Grid service providers (GSPs). The Grid user wants to make use of Grids to complete their applications. Refactoring existing applications is thus essential to ensure that these applications are Grid-enabled to run on Grids [26]. The Grid user also needs to express the service requirements to be fulfilled by GSPs. Varying QoS parameters, such as a deadline for the application to be completed and budget to be paid on completion, are defined by different Grid users,

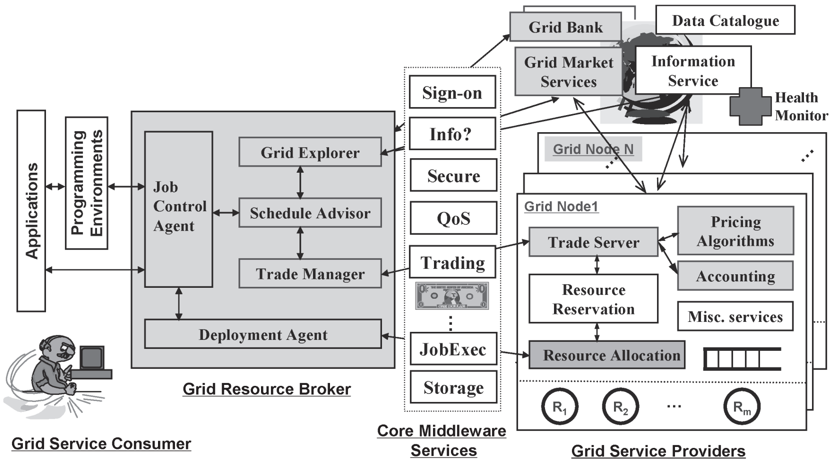

Figure 1.4 A reference service-oriented architecture for utility Grids [1].

thus resulting in dynamic fluctuation of peak and nonpeak service demands. The Grid resource broker then discovers appropriate Grid middleware services on the basis of these service demand patterns and QoS requirements, and dynamically schedules applications on them at runtime, depending on their availability, capability, and costs. A GSP needs tools and mechanisms that support pricing specifications and schemes so they can attract users and improve resource utilization. They also require protocols that support service publication and negotiation, accounting, and payment.

The Grid resource broker comprises the following components:

- *Job control agent*—interacts with users to determine the application profile that is used to create jobs that are executed on the Grid resources. These jobs are then input to the schedule advisor. The control agent also ensures that the jobs are persistent and their status is properly maintained.
- *Grid explorer*—discovers computational and data resources, including their current and future status, and also their prices.
- *Schedule advisor*—receives the list of jobs from the control agent and the list of resources from the Grid explorer. It uses these inputs to assign jobs to the suitable Grid resources such that the users' requirements are satisfied. It keeps track of the progress of execution and makes changes in the schedule accordingly.
- *Trade manager*—negotiates with resources and services selected by the schedule advisor and forms SLAs with them using pricing information gathered from the market directory and past history. It then monitors the compliance of these agreements.
- *Deployment agent*—executes jobs on the selected resources according to the assignment performed by the schedule advisor. If any advance reservations have been negotiated by the trade manager, this component also claims the nodes allocated before submitting jobs. It monitors job execution and reports job status periodically.

Traditional core Grid middleware focuses on providing infrastructure services for secure and uniform access to distributed resources. Supported features include security, single signon, remote process management, storage access, data management, and information services. An example of such middleware is the Globus toolkit [27], which is a widely adopted Grid technology in the Grid community. Market-oriented Grids require additional service-driven Grid middleware infrastructure, which includes

- *Grid market directory*—allows GSPs to publish their services so as to inform and attract users.
- *Trade server*—negotiates with Grid resource broker on the basis of pricing algorithms set by the GSP and sells access to resources by recording resource usage details and billing the users according to the agreed on pricing policy.
- *Pricing algorithms*—specifies prices to be charged to users on the basis of the GSP's objectives, such as maximizing profit or resource utilization at varying times and for different users.

- *Accounting and charging*—records resource usage and bills the users according to the terms negotiated and agreed to between Grid resource broker and trade server.

An end-to-end realization of market-oriented Grid supported by the Gridbus middleware is discussed in Chapter 26.

1.8 OPERATIONAL FLOW IN A MARKET-ORIENTED GRID

Figure 1.5 shows how services are assembled on demand in a market-oriented Grid. The application code is the legacy application to be run on the market-oriented Grid. Users first compose their application as a distributed application such as a parameter sweep using visual application composer tools (step 1). The parameter sweep model creates multiple independent jobs, each with a different parameter. This model is well suited for Grid computing environments wherein challenges such as load volatility, high network latencies, and high probability of individual node failures make it difficult to adopt a programming approach that favors tightly coupled systems. Accordingly, a parameter sweep application has been termed as a "killer application" for the Grid [28].

Visual tools allow rapid composition of applications for Grids by hiding the associated complexity from the user. The user's analysis and QoS requirements are submitted to the Grid resource broker (step 2). The Grid resource broker first discovers

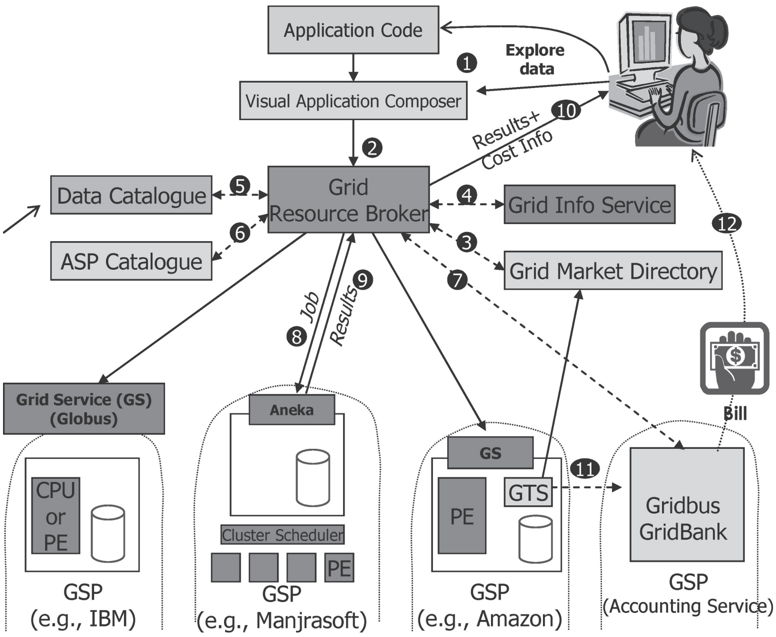

Figure 1.5 On-demand assembly of services in a utility Grid.

suitable Grid service providers (step 3) and capabilities of their services to meet QoS requirements of users, including price (step 4) using the Grid Market Directory and Grid Information Services, respectively. The broker then identifies the list of data sources or replicas through a data catalog and selects the optimal ones (step 5). The broker also identifies the list of GSPs that provides the required application services using the application service provider (ASP) catalog (step 6). The broker checks that the user has the necessary credit or authorized share to utilize the requested Grid services (step 7). The broker scheduler assigns and deploys jobs to Grid services that meet user QoS requirements (step 8). The broker agent on the Grid resource at the GSP then executes the job and returns the results (step 9). The broker consolidates the results before passing them back to the user (step 10). The metering system charges the user by passing the resource usage information to the accounting service (step 11). The accounting service reports remaining resource share allocation and credit available to the user (step 12).

1.9 MARKET-ORIENTED SYSTEMS IN PRACTICE

Economics has a long history both as a set of mechanisms by which human commerce is enacted and as a field of study involving analysis and research of these mechanisms of commerce (resource allocation and management). As these mechanisms have evolved with society, they tend to be resilient and flexible, and offer much that is directly applicable for use in computer systems. Various economic theories and models, including micro- and macroeconomic principles that can be applied for Grid computing [16], include the following:

- Commodity market models
- Posted-price models
- Bargaining models
- Tendering or contract-net models
- Auction models
- Bid-based proportional resource-sharing models
- Cooperative bartering models
- Monopoly and oligopoly

Various criteria used for judging the effectiveness of a market model [55] are social welfare (global good of all), Pareto efficiency (global perspective), individual rationality (better off by participating in negotiation), stability (mechanisms that cannot be manipulated, i.e., that behave in the desired manner), computational efficiency (protocols should not consume too much computation time), and distribution and communication efficiency (communication overhead to capture a desirable global solution).

Several research systems (see Table 1.1) have explored the use of different economic models for trading resources to manage resources in different application

TABLE 1.1 Computational Economy–Based Distributed Resource Management Systems

System Name	Economic Model	Platform	Remarks
Mariposa [47] (UC Berkeley)	Bidding (tendering/contract-net); pricing based on load and historical information	Distributed database	Supports budget-based query processing and storage management
Mungi [48] (Univ. New South Wales)	Commodity market (renting storage space that increases as available storage runs low, forcing users to release unneeded storage)	Storage servers	Supports storage objects based on bank accounts from which rent is collected for storage occupied by objects
Popcorn [40] (Hebrew Univ.)	Auction (highest bidder gets access to resource and it transfers credits from buyer to the seller account)	Web browsers (*popcorn parallel code* runs within a browser of provider)	Popcorn API-based parallel applications need to specify a budget for processing each of its modules
Java Market [41] (Johns Hopkins Univ.)	QoS-based computational market [resource owner receives $f(j,t)$ award for completing f in time t]	Web browsers (JavaMarket runs *standard Java applets* within a browser)	One can sell CPU cycles by pointing Java-enabled browser to portal and allow execution of applets
Enhanced MOSIX [42]	Commodity market (resource cost of each node is known)	Clusters of computers (Linux PCs)	Supports process migration such that overall cost of job execution is kept low
Xenoservers [44] (Cambridge Univ.)	Bidding (proportional resource sharing)	Single computer	Accounted execution of untrusted code
D'Agents [45] (Dartmouth College)	Bidding (proportional resource sharing)	Single computer or mobile agents	Agent's bid function is proportional to benefit
Rexec/Anemone [46] (UC Berkeley)	Bidding/auction (for proportional resource sharing)	Clusters (a market-based cluster batch queue system)	Users assign utility value to their application and system allocates resources proportionally
Spawn [39] (Xerox PARC)	Second-price/Vickery auction (uses sponsorship model for funding money to each task depending on some requirements)	Network on workstations; each WS executes a single task per time slice	Supports execution of concurrent program expressed in the form of hierarchy of processes that expand and shrink size depending on the resource cost

CSAR supercomputing center [52] (Manchester)	Commodity market and priority-based model (they charge for CPU, memory, storage, and human support services)	MPPs, Crays, clusters, and storage servers	Any application can use this service, and QoS is proportional to user priority and scheduling mechanisms
Nimrod-G [15] (Monash Univ.)	Supports economy models such as commodity market, spot market, and contract-net for price establishment	Worldwide Grid (resources Grid-enabled using Globus middleware)	A *real* system that supports deadline and budget constrained algorithms for scheduling task-farming and data parallel applications on global Grids.
G-Commerce [49] (UC Santa Barbara)	Commodity and auctions	Simulates hypothetical consumers and produces	Explores strategies for pricing Grid resources to enable resource trading
OCEAN [51] (Univ. Florida)	Continuous double auction	A Java-based platform with distributed PCs	Explores the use of continuous double auction for trading computational resources
Stanford Peers [49] (Stanford Univ.)	Auctions with cooperative bartering in a cooperative sharing environment	Simulates storage trading for content management	Demonstrates distributed resource trading policies based on auctions by *simulation*
Gridbus [54] (Univ. Melbourne)	Both commodity and auction models for clusters, computational and data Grids	Clusters, clouds, and Grids	Provides services for ensuring end-to-end QoS at various levels for users in cluster and Grid environment
CatNets [57] (Europe)	Interconnected markets of providers and consumers	Grid and service-oriented computing	The approach is to have free markets of participants wherein consumers procure services on the basis of dynamic price changes caused by market processes.
SORMA [59] (Europe)	Market-based system consisting of self-interested resource brokers and user-agents	Open Grid market on top of existing resource management systems	The open Grid market matches users and brokers; resource allocation is mediated via SLAs
GridEcon [60] (Europe)	Markets for trading of resources between small and medium enterprises	Resources managed by commercial service providers	At the time of writing, the project is researching into a viable business model for trading services in an open market

domains: CPU cycles, storage space, database query processing, and distributed computing. They include Spawn [39], Popcorn [40], JavaMarket [41], Enhanced MOSIX [42], JaWS [43], Xenoservers [44], D'Agents [45], Rexec/Anemone [46], Mariposa [47], Mungi [48], Stanford Peers [49], G-Commerce [50], OCEAN [51], Nimrod/G [15], Gridbus [54], CatNets [57], SORMA [59], and GridEcon [60]. These systems have been targeted to manage single or multiple resources for application domains as follows:

- *Single-domain computing systems*—enhanced MOSIX and Rexec/Anemone
- *Agent-based systems*—Xenoservers and D'Agents
- *Distributed database management system*—Mariposa
- *Shared storage management system*—Mungi
- *Storage space trading system*—Stanford Peers
- *Web-based distributed systems*—Popcorn and Java Market
- *Multidomain distributed Grid systems*—Nimrod-G and Gridbus Broker

Many of the resource management systems presented in Table 1.1 follow a single model for resource trading. They have been designed with a specific goal in mind, for either CPU or storage management. In order to use some of these systems, applications have to be designed using their proprietary programming models, which is generally discouraging, as applications need to be specifically developed for executing on those systems. Also, resource trading and job management modules have been developed using monolithic system architecture that limits their extensibility.

Multidomain Grid systems such as Nimrod/G and Gridbus brokers separated these two concerns through a layered design approach to support different middleware technologies that coexist with trading strategies and user-level resource brokers. The resource trading services are offered as core services and can be used by different higher-level services such as resource brokers and resource-aware applications.

Typically, in a Grid marketplace, the resource owners and users can use any one or more of these models or even combinations of them in meeting their objectives [14]. Both have their own expectations and strategies for being part of the Grid. The resource consumers adopt the strategy of solving their problems at a low cost within a required timeframe. The resource providers adopt the strategy of obtaining the best possible return on their investments while trying to maximize their resource utilization by offering a competitive service access cost in order to attract consumers. The resource consumers can choose providers that best meet their requirements.

Both GRBs and GSPs can initiate resource trading and participate in the interaction depending on their requirements and objectives. GRBs may invite bids from a number of GSPs and select those that offer the lowest service costs and meet their deadline and budget requirements. Alternatively, GSPs may invite bids in an auction and offer services to the highest bidder as long as its objectives are met. Both GSPs and GRBs have their own utility functions that must be satisfied and maximized. The GRBs perform a cost–benefit analysis depending on the deadline (by which the results are required) and budget available (the amount of money that the user is willing to invest for solving the problem). The resource owners decide their pricing on the basis of various factors. They may charge different prices for different users using the same service depending on the specific user demands. Resources may have different prices on the basis of environmental influences such as the availability of larger core memory and better communication bandwidth with the outside world.

Grid brokers (note that each user has his/her own broker as his/her agent in a Grid environment) may have different goals (e.g., different deadlines and budgets), and each broker tries to maximize its own good without concern for the global good. This needs to be considered in building automated negotiation infrastructure. In a *cooperative distributed computing or problem-solving environment* (like a cluster of computers), the system designers impose an *interaction protocol* (possible actions to take at different points) and a *strategy* (a mapping from one state to another and a way to use the protocol). This model aims for global efficiency as nodes cooperate toward a common goal. On the other hand, in Grid systems, brokers and GSPs are provided with an interaction protocol, but they choose their own private strategy (as in multiagent systems), which cannot be imposed from outside. Therefore, the negotiation protocols need to be designed assuming a *noncooperative, strategic* perspective. In this case, the main concern is what social outcomes follow given a protocol, which guarantees that each broker/GSP's desired local strategy is best for her/him and hence that she/he will use it.

1.10 CHALLENGES OF UTILITY COMPUTING MODELS FOR GRIDS

The adoption of utility computing requires a change in the manner in which many IT operations are planned and performed in the enterprise. This induces many challenges that need to be addressed both technologically and culturally. One challenge is that both providers and users need to redraft and reorganize their current IT-related procedures and operations to incorporate the usage of services from external providers [30]. New IT policies need to be negotiated and agreed on between providers and users, compared to the previous situation where providers and users owned and controlled their standalone policies. Providers must also understand specific service needs and requirements of users in order to design suitable policies for them.

The end user in an organization must not be able to perceive a difference between an internal, dedicated service and one that is sourced from an external provider. Therefore, the utility computing experience has to be seamless and should increase the flexibility of service consumption. This motivates the need for appropriate software interfaces and toolkits that enable an organization to expand and contract its service usage at will without causing disruptions in the users' workflows. Currently, toolkits such as Amazon Web Services and Google App Engine allow a measure of flexibility for external users deploying services on the companies' infrastructure. However, these are provider-specific and are short of features such as variable, demand-driven usage pricing models, and integration with business processes. True utility computing requires vendors and providers to move away from proprietary services toward open standards so that users and producers experience fewer difficulties and complexities in integrating technologies and working together, thus reducing associated costs.

As seen so far, in a utility computing scenario, the users cede much of their abilities to control dedicated resources. Therefore, the only way that they can obtain guarantees to meet their QoS requirements is through binding SLAs with service providers. Such SLAs have penalty clauses that are triggered when the guarantees are violated. Providers therefore need strategies for selecting SLAs such that the risk of violations is minimized [31,32]. Improved service-oriented policies and autonomic controls [33,34] are essential for achieving this.

Financial risk management for utility computing [35] consist of two factors: delivery risk and pricing risk. *Delivery risk* factors examine the risks concerned with each possible scenario in which a service can be delivered. *Pricing risk* factors study the risks involved with pricing the service with respect to the availability of resources. Given shorter contract durations, lower switching costs, and uncertain customer demands in utility computing environments, it is important to have dynamic and flexible pricing schemes to potentially maximize profits and minimize losses for providers [36]. Also, current providers have rigid SLAs that do not offer flexibility to allow tradeoff of lower QoS requirements for lower charges. Such ability would enable providers to supply services to satisfy a range of users, thereby increasing the profitability of their enterprise. Therefore, this motivates the adoption of different negotiation mechanisms through which providers and consumers can arrive at mutually agreeable terms in SLAs. This would also require both parties to adopt sophisticated negotiation strategies that mirror interactions between humans in real marketplaces.

There are also potential nontechnical obstacles to the successful adoption of utility computing such as cultural and people-related issues that will require organizations to change their current stance and perceptions [37]. The most worrying issues that are perceived are loss of control or access to resources, risks associated with enterprisewide deployment, loss or reduction of budget dollars, and reduced priority of projects. Also important are concerns about security, particularly with regard to maintaining the confidentiality of data while they are being processed on providers' resources. Regulations governing confidentiality and disclosure may prohibit many applications from accessing utility computing services. Ultimately, applications dealing with sensitive areas such as national security and health may be forever confined to dedicated resources. Yet, overcoming these nontechnical obstacles is extremely critical and requires the dissemination of correct information to all levels of management within organizations to prevent the formation of misperceptions.

1.11 SUMMARY AND CONCLUSION

To summarize, we have presented the emergence of Grid computing as a platform for next-generation parallel and distributed computing. We have covered some of the major Grid efforts around the world and discussed the Grid software stack with two sample technologies. We have identified various challenges in managing Grid resources owned by different organizations. We have introduced computational economy, that is, a market-oriented model for tackling challenges of resource management within large-scale Grids. We have discussed the reference architecture for market-oriented Grids and presented a scenario for its operation. Various approaches followed by representative works in market-oriented resource management and application scheduling are also discussed.

It can be observed that while significant effort has been devoted to development of Grid technologies, still more must be achieved in terms of Grids providing computing utilities in the same manner as power utilities supply electric power. The Grid resource management systems must dynamically trade for the best resources according to a metric of the price and performance available, and schedule computations on these resources such that they meet user requirements. The Grid middleware needs to offer services that help resource brokers and resource owners to trade for resource access. Market-based Grid systems need to pay attention to the reputation of service providers, where regular offenders are penalized and additional incentives are provided for good services [61]. Ultimately, this would require the development of richer services and applications on top of existing ones so that Grid computing can move beyond scientific applications into the mainstream IT infrastructure.

ACKNOWLEDGMENTS

This chapter is partially derived from the authors' past work, especially from the PhD thesis [1] of the first author. We would like to thank various colleagues, especially David Abramson (Monash University), with whom we have investigated market-oriented Grid computing. We would also like to thank Chee Shin Yeo for his comments on this chapter.

REFERENCES

1. R. Buyya, *Economic-Based Distributed Resource Management and Scheduling for Grid Computing*, PhD thesis, Monash Univ., Melbourne, Australia, April 2002.
2. B. Chun, *Market-Based Cluster Resource Management*, PhD dissertation, Univ., California, Berkeley, CA, Oct. 2001.
3. L. Kleinrock, A vision for the Internet, *ST Journal of Research* **2**(1):4–5 (Nov. 2005).
4. I. Foster and C. Kesselman, eds., The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann, San Francisco, 1999.
5. T. Hey and A. Trefethen, The UK e-Science Core Programme and the Grid, *Journal of Future Generation Computer Systems* **18**(8):1017–1031 (2002).
6. M. Chetty and R. Buyya, Weaving computational grids: How analogous are they with electrical Grids?, *Computing in Science and Engineering* **4**(4):61–71 (July–Aug. 2002).
7. F. Berman, G. Fox, and A. Hey, eds., Grid Computing: Making the Global Infrastructure a Reality, Wiley, Hoboken, NJ, 2003.

8. M. Baker, R. Buyya, and D. Laforenza, Grids and Grid technologies for wide-area distributed computing, *Software: Practice and Experience* **32**(15):1437–1466 (Dec. 2002).
9. I. Foster, C. Kesselman, and S. Tuecke, The anatomy of the Grid: Enabling scalable virtual organizations, *International Journal of High Performance Computing Applications*, **15**(3):200–222 (2001).
10. R. Buyya, D. Abramson, and J. Giddy, An economy driven resource management architecture for global computational power Grids, *Proc. 7th International Conf. Parallel and Distributed Processing Techniques and Applications*, Las Vegas, June 26–29, 2000.
11. W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, and K. Stockinger, Data management in an international data Grid project, *Proc. 1st IEEE/ACM International Workshop on Grid Computing, Bangalore, India, Dec*. 2000.
12. Open Grid Forum, http://www.ogf.org.
13. M. Rappa, The utility business model and the future of computing services, *IBM Systems Journal* **43**(1):32–42 (2004).
14. R. Buyya, D. Abramson, and J. Giddy, A case for economy Grid architecture for service-oriented Grid computing, *Proc. IPDPS, 10th IEEE International Heterogeneous Computing Workshop*, San Francisco, CA, April 23, 2001.
15. R. Buyya, D. Abramson, and J. Giddy, Nimrod-G: An architecture for a resource management and scheduling system in a global computational Grid, *Proc. 4th International Conf. High Performance Computing in Asia-Pacific Region, Beijing, China, May* 2000.
16. R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, Economic models for resource management and scheduling in Grid computing, *Concurrency and Computation: Practice and Experience* **14**(13–15):1507–1542 (Nov.–Dec. 2002).
17. C. Catlett, P. Beckman, D. Skow, and I. Foster, Creating and operating national-scale cyberinfrastructure services, *CTWatch Quarterly* **2**:2–10 (2006).
18. W. T. Sullivan, III, D. Werthimer, S. Bowyer, J. Cobb, D. Gedye, and D. Anderson, A new major SETI project based on Project Serendip data and 100,000 personal computers, *Proc. 5th International Conf. Bioastronomy*, 1997.
19. R. Buyya, Grid and peer-to-peer (P2P) commercial companies, *Grid Computing Information Centre*, http://www.gridcomputing.com/, 2000–2008.
20. ISO New England, Electricity trading over the Internet begins in six New England states, *Business Wire*, http://industry.java.sun.com/javanews/stories/story2/0,1072,15093,00.html, May 13, 1999.
21. I. Foster, A. Roy, and V. Sander, A quality of service architecture that combines resource reservation and application adaptation, *Proc. IEEE/IFIP 8th International Workshop on Quality of Service* (*IWQOS 2000*), Pittsburgh, PA, June 2000.
22. R. Cocchi, S. Shanker, D. Estrin, and L. Zhang, Pricing in computer networks: Motivation, formulation, and example, *IEEE/ACM Transactions on Networking* **1**(6):614–627 (Dec. 1993).
23. A. Lazar and N. Semret, *Auctions for Network Resource Sharing*, Technical Report TR 468-97-02, Columbia Univ., Feb. 1997.
24. L. W. McKnight and J. Boroumand, Pricing Internet services: Approaches and challenges, *IEEE Computer* **33**(2):128–129 (Feb. 2000).

25. B. Carpenter, IPv6 and the future of the Internet, *The Internet Society Member Briefing*, July 23, 2001. http://www.isoc.org/briefings/001/.
26. IBM developerWorks, Six strategies for Grid application enablement, http://www.ibm.com/developerworks/grid/library/gr-enable (accessed Nov. 2006).
27. The Globus Alliance, The Globus toolkit, http://www.globus.org/toolkit (accessed Nov. 2006).
28. D. Abramson, J. Giddy, and L. Kotler, High performance parametric modeling with Nimrod/G: Killer application for the global Grid?, *Proc. 14th International Parallel and Distributed Processing Symp.* (*IPDPS 2000*), Cancun, Mexico, May 2000.
29. Y. Tanaka, H. Nakada, S. Sekiguchi, T. Suzumura, and S. Matsuoka, Ninf-G: A reference implementation of RPC-based programming middleware for Grid computing *Journal of Grid Computing* **1**(1):41–51 (2003).
30. R. Buyya, T. Cortes, and H. Jin, Single system image, *International Journal of High Performance Computing Applications* **15**(2):124–135 (summer 2001).
31. M. Buco, R. Chang, L. Luan, C. Ward, J. Wolf, and P. Yu, Utility computing SLA management based upon business objectives, *IBM Systems Journal*, **43**(1):159–178 (2004).
32. C. S. Yeo and R. Buyya, Integrated risk analysis for a commercial computing service, *Proc. 21st IEEE International Parallel and Distributed Processing Symp.* (*IPDPS 2007*), Long Beach, CA, March 26–30, 2007.
33. J. Kephart and D. Chess, The vision of autonomic computing, *IEEE Computer* **36**(1):41–50 (Jan. 2003).
34. R. Murch, Autonomic Computing, Prentice-Hall, 2004.
35. C. Kenyon and G. Cheliotis, Elements of financial risk management for Grid and utility computing, in *Handbook of Integrated Risk Management for E-Business: Measuring, Modeling, and Managing Risk*, A. Labbi, ed., J. Ross Publishing, 2005.
36. G. Paleologo, Price-at-risk: A methodology for pricing utility computing services, *IBM Systems Journal* **43**(1):20–31 (2004).
37. Platform computing, the politics of Grid, http://www2.platform.com/adoption/politics (accessed Nov. 2006).
38. J. Basney and M. Livny, Deploying a high throughput computing cluster, *High-Performance Cluster Computing*, R. Buyya, ed., Prentice-Hall PTR, May 1999.
39. C. Waldspurger, T. Hogg, B. Huberman, J. Kephart, and W. Stornetta, Spawn: A distributed computational economy, *IEEE Transactions on Software Engineering* **18**(2):103–117 (Feb. 1992).
40. N. Nisan, S. London, O. Regev, and N. Camiel, Globally distributed computation over the Internet: The POPCORN project, *Proc. 18th International Conf. Distributed Computing Systems*, Amsterdam, The Netherlands, May 26–29, 1998.
41. Y. Amir, B. Awerbuch, and R. Sean Borgstrom, A cost-benefit framework for online management of a metacomputing system, *Proc. 1st International Conf. Information and Computational Economy*, Charleston, VA, Oct. 25–28, 1998.
42. Y. Amir, B. Awerbuch., A. Barak A., S. Borgstrom, and A. Keren, An opportunity cost Approach for job assignment in a scalable computing cluster, *IEEE Transactions on Parallel and Distributed Systems* **11**(7):760–768 (July 2000).
43. S. Lalis and A. Karipidis, An open market-based framework for distributed computing over the Internet, *Proc. 1st IEEE/ACM International Workshop on Grid Computing* (*Grid 2000*), Bangalore, India, Dec. 17, 2000.

44. D. Reed, I. Pratt, P. Menage, S. Early, and N. Stratford, Xenoservers; accounted execution of untrusted code, *Proc. 7th Workshop on Hot Topics in Operating Systems* (*HotOS-VII*), Rio Rico, AZ, March 28–30, 1999.

45. J. Bredin, D. Kotz, and D. Rus, *Utility Driven Mobile-Agent Scheduling*, Technical Report CS-TR98-331, Dartmouth College, Hanover, NH, Oct. 3, 1998.

46. B. Chun and D. Culler, *Market-Based Proportional Resource Sharing for Clusters*, Technical Report CSD-1092, Univ. California, Berkeley, Jan. 2000.

47. M. Stonebraker, R. Devine, M. Kornacker, W. Litwin, A. Pfeffer, A. Sah, and C. Staelin, An economic paradigm for query processing and data migration in Mariposa, *Proc. 3rd International Conf. Parallel and Distributed Information Systems*, Austin, TX, Sept. 28–30, 1994.

48. G. Heiser, F. Lam, and S. Russell, Resource management in the Mungi single-address-space operating system, *Proc. Australasian Computer Science Conf., Perth, Australia, Feb. 4–6*, 1998.

49. B. Cooper and H. Garcia-Molina, Bidding for storage space in a peer-to-peer data preservation system, *Proc. 22nd International Conf. Distributed Computing Systems* (*ICDSC 2002*), Vienna, Austria, July 2–5, 2002.

50. R. Wolski, J. Plank, J. Brevik, and T. Bryan, Analyzing market-based resource allocation strategies for the computational grid, *International Journal of High-Performance Computing Applications* **15**(3):258–281 (fall 2001).

51. P. Padala, C. Harrison, N. Pelfort, E. Jansen, M. Frank, and C. Chokkareddy, OCEAN: The Open Computation Exchange and Arbitration Network, a market approach to meta computing, *Proc. 2nd International Symp. Parallel and Distributed Computing, Ljubljana, Slovenia, Oct*. 2003.

52. J. Brooke, M. Foster, S. Pickles, K. Taylor, and T. Hewitt, Mini-grids: Effective test-beds for Grid application, *Proc. 1st IEEE/ACM International Workshop on Grid Computing* (*Grid 2000*), Bangalore, India, Dec. 17, 2000.

53. G. Allen, T. Goodale, T. Radke, M. Russell, E. Seidel, and K. Davis, Enabling applications on the grid: A Gridlab overview, *International Journal of High-Performance Computing Applications* **17**(4):449–466 (2003).

54. R. Buyya and S. Venugopal, The Gridbus toolkit for service oriented Grid and utility computing: An overview and status report, *Proc. 1st IEEE International Workshop on Grid Economics and Business Models* (*GECON 2004*), Seoul, South Korea, April 23, 2004.

55. T. Sandholm, Distributed rational decision making, in *Multi-Agent Systems: A Modern Introduction to Distributed Artificial Intelligence*, G. Weiss, ed., The MIT Press, Cambridge, MA, 2000.

56. R. Buyya, D. Abramson, and S. Venugopal, The Grid economy, *Proceedings of the IEEE* **93**(3):698–714 (March 2005).

57. T. Eymann, M. Reinicke, et al., Catallaxy-based Grid markets, *Multiagent and Grid Systems* **1**(4):297–307 (2005).

58. A. Rajasekar, M. Wan, R. Moore, and W. Schroeder, A prototype rule-based distributed data management system, *Proc. Workshop on Next Generation Distributed Data Management: 15th ACM/IEEE International Symp. High Performance Computing, Paris, May* 2006.

59. D. Neumann, J. Stoesser, A. Anandasivam, and N. Borissov, SORMA—building an open Grid market for Grid resource allocation, *Proc. 4th International Workshop of Grid Economics and Business Models* (*GECON 2007*), Rennes, France, Aug. 28, 2007.

60. J. Altmann, C. Courcoubetis, J. Darlington, and J. Cohen, GridEcon–the economic-enhanced next-generation Internet, *Proc. 4th International Workshop of Grid Economics and Business Models* (*GECON 2007*), Rennes, France, Aug. 28, 2007.

61. J. Broberg, S. Venugopal, and R. Buyya, Market-oriented Grids and utility computing: The state-of-the-art and future directions, *Journal of Grid Computing* **6**(3):255–276 (2008).