

# Part



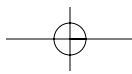
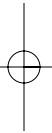
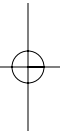
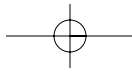
# Theory and Foundations

This part covers theory describing all the major initiatives in distributed and grid computing along with market reports and segmentation data, including the following:

**Overview of High Performance Computing (HPC):** Chapter 1 describes terminology used in distributed and grid computing and gives a quick summary of the evolution of grid technologies. It also describes current grid initiatives in science and technology.

**Enterprise Computing:** Chapter 2 explains distributed computing concepts focusing on the enterprise or virtual organization. Concepts such as infrastructures, shared services, business components, client state, pooled resources, and database concurrency. All these concepts play a role in today's competitive IT organization. Also in Chapter 2 is a discussion of computational economy, which is an emerging field in distributed computing. Its main goal is to drive grid technologies into the mainstream business world. This chapter describes exciting research conducted in the fields of resource management models, computational economy, and economy-driven grid applications on the enterprise.

**Core Grid Middleware:** Chapter 3 describes some of the middle tier technologies used on grid computing, including peer-to-peer (P2P) architectures, the Globus Toolkit, and research conducted on grid and business fields.



# 1 The Roadmap to High-Performance Computing

## In This Chapter

- Evolution of Grid Technologies
- The Grid in a Nutshell
- Distributed Computing Models
- Computing on Demand (COD)
- Grids in Science and Technology
- Summary
- References

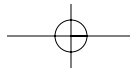
This chapter provides background information on the evolution of high-performance computing (HPC) from the early decades of networked computers to the latest technological advances in distributed systems. This chapter includes the following topics:

- A brief history of HPC
- An overview of the concept of the grid
- Explanations of the latest distributed computing models such as Internet computing, peer-to-peer, and grid architectures
- A brief discussion of some of the latest grid projects in science and technology

## 1.1 EVOLUTION OF GRID TECHNOLOGIES

---

High-performance computing has its roots in the early 1940s with the *Manhattan Project* and early efforts by the Department of Energy (DOE) to develop advanced



## 4 Grid Computing for Developers

computing capabilities to solve critical problems of interest. In those days parallel computing was done by mathematicians and the type of problems that could be solved was very limited. Many algorithms and computational methods developed by the DOE are still used today. For example, the *Monte Carlo method*, where statistical samples are used to predict behaviors of a large group, was developed by John von Neumann and others in 1946, and it is still used today in stock market forecasting, medicine, traffic flow, and others fields.

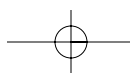
In the early 1950s, researchers pushed the state of the art computing to the limits by building their own computers and looking into commercial models. Thanks to the groundbreaking work of mathematicians like Alan Turing, ideas on machine intelligence were developed.

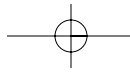
Major milestones of the 1950s are the following:

- Vacuum tubes became a thing of the past to open the way for the transistor allowing for the mass production of computers.
- The invention of the modem allowed scientists to access systems remotely.
- The Department of Defense created ARPANET, the first nationwide network connecting labs together and allowing researches access to a wide area network of computing systems.

Also in that decade, computers were mostly built to solve specific applications. Companies pushed the bounds of computational research to the limits in their effort to achieve greater memory density. Physics discoveries such as the ion-channeling effect allowed chip manufacturers to draw transistors inside blocks of silicon. More powerful computers provided broader support for energy research in applications such as fusion plasma modeling and atmosphere and emissions modeling.

In the 1970s, computers were first linked by networks, thus giving birth to the idea of harnessing idle computing power. Scientists at the Xerox Palo Alto Research Center (PARC) created the first Ethernet network beginning the first experiments on distributed computing. Scientists John F. Shoch and Jon Hupp developed the famous Internet worm, which was designed to move around using idle CPU cycles for beneficial purposes. Software to perform computations and cooperate with other machines on the network was developed. In the early 1980s, more powerful computers were developed, leaving researches with the task of writing their own operating systems. Advances such as *timesharing* (sharing computing power by multiple simultaneous users) were developed for Cray computers. NFSnet (that will later become the Internet) took center stage after researchers realized the critical role of networking on scientific computing. A common filesystem was developed, allowing remote computers to share storage resources. The first multiprocessor vector computers were built, allowing researchers to work on more complex problems using parallel systems.





The evolution of the Internet in the 1990s saw the creation of two groundbreaking distributed projects: The first was distributed.net, which used thousands of computers around the world to break encryption codes. The second, and one of the most successful, is the popular SETI@home project [SETI]. The goal of SETI is to look for radio signal fluctuations that could indicate a signal from intelligent life emanating from outer space. It originated at the University of California at Berkeley in 1999, becoming the most successful Internet distributed project with more than two million volunteers installing the SETI software agent.

As we jump into the 21st century, high-performance computing plays a phenomenal role in scientific advances: environmental simulations, unlocking the genetic code, exploring the basic structure of matter and the universe. All these accomplishments have been made possible through the advances in computing and simulation science. But we have only begun to grasp the future of what is to come in the next century!

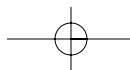
## **1.2 THE GRID IN A NUTSHELL**

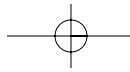
---

As industry has become involved in distributed systems, the term *grid* has become a marketing slogan—so much so that any type of distributed filesystem can be called a storage grid, or a scheduler deployed in a cluster can be called a cluster grid, or if a user connects through a file-sharing application, he could be using a digital media distribution grid. Experts suggest that a grid must be evaluated for the applications, business value, and scientific results that it delivers, rather than for its architecture. Carl Kesselman [GridAnatomy01] defined the grid as a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities. This definition suggests the idea of a form of on-demand access to computing, data, and services that evolved from early ideas of computer utilities. Indeed, in the 1960s, Len Kleinrock suggested the spread of computer utilities, similar to electric and telephone utilities that will service people [FosterChecklist02].

Foster and colleagues suggested addressing social and policy issues with the idea of coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations [GridAnatomy01]. This implies resource-sharing arrangements between providers and consumers resulting in a common purpose. This sharing of computers, software, data, and other resources is highly controlled by resource providers and consumers defining clearly what is shared, who is allowed, and the conditions under which sharing occurs. A logical group of resource providers and consumers is known as a virtual organization (VO) [FosterChecklist02].

Foster suggests the following checklist for characteristics of a grid-enabled application:





## 6 Grid Computing for Developers

1. *Decentralized resource coordination*: A grid should integrate and coordinate resources and users in different domains and address issues of security, policy, payment, and membership.
2. *Standards and open source protocols*: These should be used for authentication, authorization, resource discovery, and resource access.
3. *Quality of service delivery*: Resources should be used in a coordinated fashion to deliver quality of service, response times, throughput, and availability to meet complex user demands.

### 1.2.1 What Can Be Called a Grid Application

According to this checklist, scheduler software such as Portable Batch System (PBS) or Sun Grid Engine (SGE), even though they deliver quality of service, don't constitute a grid because of their centralized host management policies. According to Foster and Kesselman, the Web is not yet a grid itself even though it is built on top of open, general-purpose protocols for access to distributed resources; it does not coordinate the use of those resources to deliver quality of service. Lately, schedulers such as Platform's MultiCluster can be called grids, as can distributed computing systems such as Condor®, SETI, and United Devices, which harness idle desktops; peer-to-peer (P2P) systems such as Gnutella, which support file sharing among participating peers, and the Storage Resource Broker, which supports distributed access to data resources [FosterChecklist02].

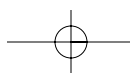
## 1.3 DISTRIBUTED COMPUTING MODELS

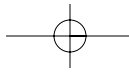
---

Modern distributed computing frameworks can be classified in three distinct branches: Internet computing, which seeks to harness idle CPU cycles; peer-to-peer for serverless communication; and grid to bridge the gap among client/server and Web Services. The following sections introduce concepts, similarities, and differences among these.

### 1.3.1 Internet Computing

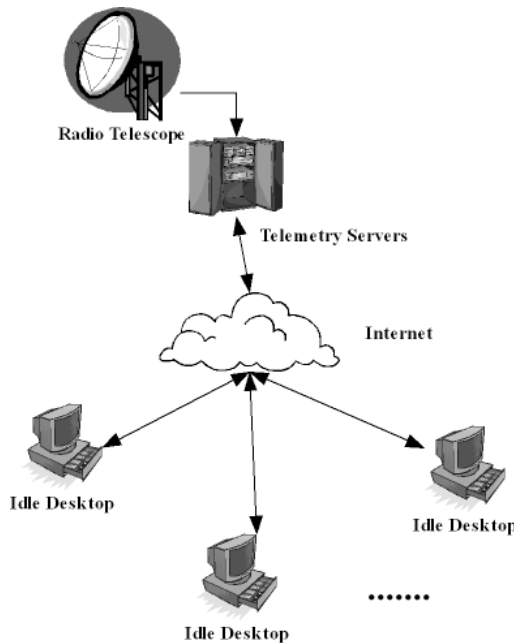
Decentralized distributed models seek to harness the computing power of millions of devices worldwide, such as personal computers or pervasive devices such as personal digital assistants (PDAs), laptops, and others. Large-scale experiments such as Search for Extraterrestrial Intelligence (SETI) launched this model into the mainstream. By collecting radio telescope data delivered on tapes to SETI@home headquarters in Berkeley, California, researchers chop the raw data into small "work-units." These work-units are then distributed to users around the world, who analyze the data on their PCs.





This model is useful in environments where thousands and possibly millions of pervasive nodes, nodes with relatively low processing power, work in a transient manner. This basically means that users will be online-offline, connecting or disconnecting all the time. This method has the advantage of achieving a degree of sensitivity that could only be dreamed of by other methods such as mainframe processes. It also scales massively as millions of nodes connect and disconnect many times, generating low network latency.

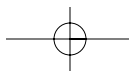
On the other hand, the public nature of this model can raise serious security issues and can be prone to malicious attacks. This model also lacks advanced clustering features such as time sharing, resource management and brokering, information services, and search capabilities (see Figure 1.1).

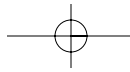


**FIGURE 1.1** SETI@home architecture, a typical Internet computing application that harnesses idle desktop power.

### 1.3.2 Peer-to-Peer (P2P)

The infamous Napster software is the prime example of a P2P distributed model. In a P2P network, computers are called peers and take advantage of resources: storage, cycles, content, and human presence available in the Internet. Peers access decentralized resources in an environment of unstable connectivity and unpredictable IP addresses. Thus, P2P nodes operate outside the Domain Name System (DNS).





## 8 Grid Computing for Developers

The main difference with this model is that peers have significant or total autonomy from central servers. This makes P2P unique. All distributed models seek to leverage unused resources, by aggregating cycles, sharing digital content, and working with the variable connectivity of possibly millions of devices.

### 1.3.2.1 Advantages and Disadvantages

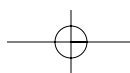
An important goal in P2P networks is that the bandwidth of clients is aggregated, so the available download bandwidth for a given user grows with the number of nodes. Another advantage includes leveraging unused resources. Disadvantages include very high network latency, inefficient search mechanisms, and although peers claim to have autonomy from central servers, applications such as Napster and Kazaa rely on a centralized indexing server to store metadata about peers. P2P applications have caused serious legal controversies also by violating copyright laws on digital media distribution.

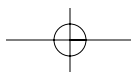
Besides file sharing, P2P is gaining ground in the online gaming industry and collaboration with applications such as Groove and others.

### 1.3.3 Grid Architectures

According to Foster and colleagues, three characteristics define a grid application: It should provide decentralized resource coordination, addressing issues of security, policy, payment, and membership. It should be based on standards and open source protocols for authentication, authorization, resource discovery, and access. Finally, it should deliver quality of service by using resources in a coordinated fashion, by providing low response times, and high throughput to meet user demands. A grid architecture should be extensible and an open structure designed to solve key VO requirements. Foster and colleagues suggest the following grid architecture, defined by a set of layers (from the bottom to the top) [GridAnatomy01]:

**Fabric:** This is the bottom layer and consists of resource and connectivity protocols, which facilitate the sharing of individual resources. This layer provides the resources (computational, storage systems, catalogs, and network resources) shared access mediated by grid protocols. Richer fabric functionality provides more sophisticated sharing operations. On the other hand, a simpler fabric simplifies the deployment of grid infrastructure. An example of this is advance reservation, which allows scheduling of resources in ways otherwise impossible to achieve. However, advance reservation increases the cost of incorporating new resources into a grid. At a minimum, the fabric layer should implement enquiry mechanisms for discovery of their structure, state, and capabilities, and resource management mechanisms to deliver quality of service. The types of resources manipulated by the fabric can be computational, storage, network, code repositories, and databases.





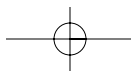
**Connectivity:** This layer defines communication and authentication protocols for network transactions. The goal is to provide easy and secure communications. Communication protocols include Internet (IP), transport (TCP, UDP), and application (DNS, and so on), with space for new protocols as the need arises. Authentication protocols should be able to provide the following: *single sign on* (log in once and access multiple resources defined by the fabric layer); *delegation*, to allow a program to run on the user's behalf so it is able to access the resources on which the user is authorized; *integration* with local security solutions; and *user-based trust relationships*, so resource providers are not required to interact before a user to access resources on either provider.

**Resource:** This layer defines protocols for secure negotiation, initiation, monitoring, control, accounting, and payment of sharing operations on individual resources. These protocols deal with individual resources and ignore global state and atomic actions across distributed collections that are handled by the collections layer. Resource layer protocols can be *information protocols* used to obtain information about configuration, load, or usage policies, and *management protocols* that negotiate access to shared resources by handling resource requirements and operation(s) to be performed. Management protocols ensure consistency of operations for a given shared resource.

**Collective:** This layer defines protocols and services global in nature and captures interactions across collections of resources. Examples of collective protocols are the following:

- a. *Directory services*: for resource properties discovery.
- b. *Coallocation, scheduling, and brokering services*: for allocation of one or more resources for a specific purpose and the scheduling of tasks.
- c. *Monitoring and diagnostics*: for failure, intrusion detection, overload, and so on.
- d. *Data replication services*: for storage management to maximize data access performance.
- e. *Grid-enabled programming systems*: to provide a programming model for resource discovery, security, allocation, and others.
- f. *Workload management systems*: for description and management of multi-component workflows.
- g. *Software discovery services*: for optimal software selection.
- h. *Community authorization servers*: to enforce community policies governing resource access.
- i. *Accounting/payment and collaboration services*.

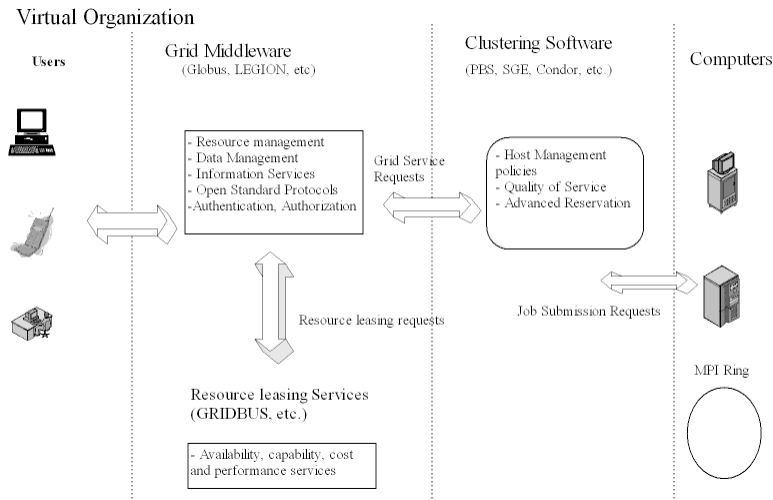
**Applications:** This layer includes user applications that operate within a VO. Applications call on services defined at any layer—resource management, data access, resource discovery, and so forth—to perform desired actions. Applications rely on application programming interfaces (APIs) implemented by software development kits (SDKs), which in turn call grid protocols to interact



## 10 Grid Computing for Developers

with network services that provide capabilities to the end user. Protocols may implement local functionality or interact with other protocols.

A typical example of a grid-enabled framework should include some of the components specified in Figure 1.2.

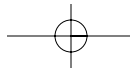


**FIGURE 1.2** The typical framework of a grid-enabled application.

### 1.3.3.1 Virtual Organizations

Two or more organizations that share resources become a VO. The policies governing access to those resources vary according to the actual organizations involved, creating an environment of providers and consumers. Resources are made available by owners with constraints on when, where, and what can be done on them. Resource consumers may also place constraints on properties of the resources they are prepared to work with. For example, a consumer may accept a resource over a secure channel only.

This environment effectively creates a dynamic sharing relationship between providers and consumers. These dynamic relationships may be defined by policies that govern access to resources. Mechanisms for discovering and characterizing the nature of the relationships are required in this environment. For example, new users should be able to discover what resources are available in the VO and the quality of those resources. Because VOs enable disparate organizations or individuals to share resources in a controlled fashion to achieve a common goal, they are emerging as fundamental entities in modern computing [GridAnatomy01].



The management of a VO sharing relationships requires a new technology. This technology has been dubbed a grid architecture, which identifies components and how those components interact with one another.

### 1.3.3.2 The Need for Standards-Based Open Middleware

Interoperability is a central issue to be addressed in networked environments. Interoperability usually means common protocols which define the mechanisms by which users and resources connect, negotiate, and establish sharing relationships. A standards-based open architecture facilitates extensibility, interoperability, and portability making it easy to define standard services. This technology and architecture is also known as middleware. Interoperability is important because without it, users and applications are forced to use bilateral sharing arrangements, with no assurance that the mechanisms used will be extensible to other parties. This makes dynamic VO formation all but impossible. Standard protocols are fundamental for general resource sharing.

## 1.4 COMPUTING ON DEMAND (COD)

---

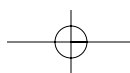
Computing on demand (COD) extends high throughput computing abilities to include a method for running short-term jobs on instantly available resources. Job management in COD-enabled systems includes interactive, compute intensive jobs, giving these jobs immediate access to the computing power they need over a relatively short period.

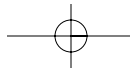
Many of the applications that are well suited for COD involve a cycle: blocking user input, computation burst to compute results, block again on user input, and so forth. When the resources are not being used for the bursts of computation to service the application, they should continue to execute long-running batch jobs. Examples of applications that require COD capabilities are the following [Condor04]:

- Graphics-rendering applications.
- Visualization tools for data mining.
- A large spreadsheet with lots of time-consuming complex formulas that take a lot of time to recalculate. When the user recalculates, the nodes work on the computation and send the results back to the master application.

### 1.4.1 How It Works

Resources on a pool of nodes run batch jobs. When a COD job appears at a node, the lower-priority (currently running) batch jobs are suspended allowing the COD job to run immediately. After completion, the batch jobs resume execution. Administra-





## 12 Grid Computing for Developers

tively, COD applications put claims on nodes. While the COD application does not need the nodes, the claims are suspended, allowing batch jobs to run [Condor04].

### 1.4.2 User Authorization

The system works by users putting claims on nodes for a COD job. A user with a claim on a resource can then suspend and resume a COD job at will. This gives the user a great deal of power on the claimed resource, even if it is owned by another user. Because of this, it is essential that users can be trusted not to abuse this power. Thus, privileges should be granted by cluster administrators and strong authentication methods used [Condor04].

### 1.4.3 Limitations

Support for COD in current schedulers faces a few limitations:

- Applications and data must be prestaged at a given node.
- Limits should be defined for how long a claim can be active, how often it is run, and so on.
- There is a lack of accounting for applications under COD claims.
- There is a lack of claim persistency on daemons.

COD provides high throughput computing abilities in an environment where short-term jobs should be run on instantly available resources. It provides computing power on demand for demanding environments such as visualization and graphics. COD capabilities are usually implemented by schedulers such as Condor [Condor04].

## 1.5 GRIDS IN SCIENCE AND TECHNOLOGY

---

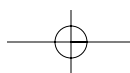
The following is a compilation of popular grid computing related consortiums and projects aimed to provide the latest technologies as well as collaboration, news, documentation, and software standards in all things *grid*.

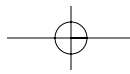
### 1.5.1 Grid Consortiums and Open Forums

The following forums contain the latest news and technologies related to grid initiatives all over the world.

#### 1.5.1.1 Global Grid Forum (GGF)—<http://www.ggf.org/>

GGF is a community-initiated forum of individuals from industry and research aiming for a global standardization of grid technologies and applications.





GGF promotes the development, deployment, and implementation of grid technologies via documentation of “best practices”—technical specifications, user experiences, and implementation guidelines.

#### **1.5.1.2 Peer-to-Peer Working Group (P2Pwg)—<http://p2p.internet2.edu/>**

P2Pwg leads efforts to investigate and explore the many aspects of peer-to-peer, beyond the resource management issues that bring the most notoriety (e.g., Internet file downloads).

Its mission is as follows [P2PInternet]:

- Report on recent occurrences and future trends within P2P and distributed computing.
- Develop collaboration between the education community and corporate entities to investigate new P2P and distributed computing applications.
- Provide best practices for resource management and P2P technologies.

#### **1.5.1.3 Asia Pacific Grid (ApGrid)—<http://www.apgrid.org/>**

ApGrid aims at building an international grid test bed among organizations in the Asia Pacific region and provides venues for sharing and exchanging ideas and information, new projects, and collaboration and interfacing with global efforts such as the GGF [ApGrid04].

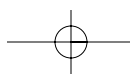
### **1.5.2 Grids in Science and Engineering**

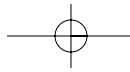
The general drive for most current grid projects is to enable the resource interactions that facilitate large-scale science and engineering projects such as bioinformatics, high-energy physics data analysis, climatology, large-scale remote instrument operation, and so forth.

In addition to government and military projects (NASA, DOE), grids are being developed by an increasing community of people who work together through coordinating organizations such as the GGF. From efforts such as this, grids will become a reality and an important component of the practice of science and engineering.

#### **1.5.2.1 The DataGrid Project—<http://eu-datagrid.web.cern.ch/eu-datagrid/>**

The DataGrid project is funded by the European Union (EU). The objective is to build the next generation computing infrastructure by providing intensive computation and analysis of shared large-scale databases, from hundreds of terabytes to petabytes, across widely distributed scientific communities. The DataGrid project will be included in the new EU grid project (Enabling Grids for E-science [EGEE]).





## 14 Grid Computing for Developers

EGEE aims to build a service grid infrastructure in Europe available to scientists 24 hours a day [EGEE04].

### 1.5.2.2 Grid Physics Network (GriPhyN)—<http://www.griphyn.org/>

The GriPhyN Project is developing grid technologies for scientific and engineering projects that must collect and analyze distributed, petabyte scale datasets. GriPhyN research will enable the development of Petascale Virtual Data Grids (PVDGs) through its virtual data toolkit (VDT). VDT is an ensemble of grid middleware that aims to make it as easy for users to deploy, maintain, and use grid middleware. VDT is composed by the following [GriPhyN05]:

- Basic grid services including Condor-G® and Globus®.
- Virtual data tools to work with virtual data, particularly the virtual data system.
- Utility software such as the Grid Security Infrastructure (GSI)-Enabled OpenSSH, software to update GSI certificate revocation lists, and monitoring software like MonaLisa.

### 1.5.2.3 Particle Physics DataGrid (PPDG)—<http://www.ppdg.net/>

The Particle Physics DataGrid Pilot (PPDG) is a collaboration of computer scientists with a strong record in grid technology and physicists with leading roles in the software and network infrastructures for major high-energy and nuclear experiments [PPDG04].

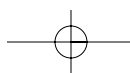
### 1.5.2.4 Petascale Data-Intensive Computing (Grid Datafarm)—<http://datafarm.apgrid.org/>

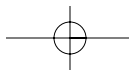
Grid Datafarm is a Petascale data-intensive computing project initiated in Japan. The project is the result of collaboration among the High Energy Accelerator Research Organization (KEK), the National Institute of Advanced Industrial Science and Technology (AIST), the University of Tokyo, and the Tokyo Institute of Technology.

The challenge involves the construction of a petascale to exascale parallel filesystems exploiting local storages of PCs spread over the worldwide grid [Grid-Farm04].

### 1.5.2.5 Resource Modeling and Simulation (GridSim)—<http://www.gridbus.org/gridsim/>

The focus of this project is to investigate effective resource allocation techniques based on computational economy through simulation. This project aims to simulate millions of resources and thousands of users with varied requirements and study the scalability of systems and algorithms, the efficiency of resource allocation policies and the satisfaction of users [GridSim04].





## 1.6 SUMMARY

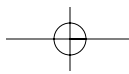
---

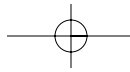
The term *grid* suggests a computer paradigm analog to a power grid. In such an environment, a shared pool of resources is created for many consumers to access as needed. Resources include processors, memory, and storage. Grid computing is still in early development, but efforts are underway to develop open standards, thus promoting its mass adoption. Major software vendors are currently working actively on those efforts.

## REFERENCES

---

- [ApGrid04] Asia Pacific Grid. National Institute of Advanced Industrial Science and Technology (AIST). May 2004. Available online from <http://www.apgrid.org/>.
- [Condor04] Condor Team. *Condor Version 6.6.6 Manual*. University of Wisconsin–Madison. Accessed online July 28, 2004, at <http://www.cs.wisc.edu/condor/manual/v6.6.6/>.
- [EGEE04] The DataGrid Project. The European Union. Available online March 2004 from <http://eu-datagrid.web.cern.ch/eu-datagrid/>.
- [FosterChecklist02] Ian Foster. *What is the Grid? A Three Point Checklist*. Argonne National Laboratory & University of Chicago, July 20, 2002.
- [GridAnatomy01] Ian Foster, Carl Kesselman, and Steven Tuecke. “The Anatomy of the Grid.” *International Journal of Supercomputer Applications*, 15(3), 200–222 (2001). Available online at [www.globus.org/research/papers/anatomy.pdf](http://www.globus.org/research/papers/anatomy.pdf).
- [GridFarm04] Grid Datafarm for Petascale Data Intensive Computing. High Energy Accelerator Research Organization (KEK), National Institute of Advanced Industrial Science and Technology (AIST), the University of Tokyo, and the Tokyo Institute of Technology. Available online November 2004 from <http://datafarm.apgrid.org/>.
- [GridSim04] Grid Computing and Distributed Systems. Department of Computer Science and Software Engineering. University of Melbourne, Australia. Available online 2004 from <http://www.gridbus.org/gridsim/>.
- [GriPhyN05] GriPhyN (Grid Physics Network). University of Florida and Argonne National Laboratory. Available online February 2005 from <http://www.griphyn.org/>.
- [P2PInternet] Peer-to-Peer Working Group. The Internet2 Consortium. Available online at <http://www.internet2.edu/>.
- [PPDG04] Particle Physics Data Grid. Argonne National Laboratory, Brookhaven National Laboratory, California Institute of Technology, Fermi National Laboratory, Lawrence Berkeley National Laboratory, San Diego Supercomputer





## 16 Grid Computing for Developers

Center, Stanford Linear Accelerator Center, Thomas Jefferson National Accelerator Facility, University of California at San Diego, University of Florida, University of Wisconsin–Madison, Harvard University, University of Manchester, University of Glasgow, State University of New York–Stony Brook, Boston University, University of Chicago, University of Texas–Arlington, University of Houston, University of Southern California Information Sciences Institute. Available online October 2004 from <http://www.ppdg.net/>.

[SETI] *The Search for Extraterrestrial Intelligence*. SETI@home. Accessed online 2001 at <http://setiathome.ssl.berkeley.edu/>.

