

THE GRID: A NEW INFRASTRUCTURE FOR 21ST CENTURY SCIENCE

As computer networks become cheaper and more powerful, a new computing paradigm is poised to transform the practice of science and engineering.

Ian Foster

Driven by increasingly complex problems and propelled by increasingly powerful technology, today's science is as much based on computation, data analysis, and collaboration as on the efforts of individual experimentalists and theorists. But even as computer power, data storage, and communication continue to improve exponentially, computational resources are failing to keep up with what scientists demand of them.

A personal computer in 2001 is as fast as a supercomputer of 1990. But 10 years ago, biologists were happy to compute a single molecular structure. Now, they want to calculate the structures of complex assemblies of macromolecules (see figure 1) and screen thousands of drug candidates. Personal computers now ship with up to 100 gigabytes (GB) of storage—as much as an entire 1990 supercomputer center. But by 2006, several physics projects, CERN's Large Hadron Collider (LHC) among them, will produce multiple petabytes (10^{15} byte) of data per year. Some wide area networks now operate at 155 megabits per second (Mb/s), three orders of magnitude faster than the state-of-the-art 56 kilobits per second (Kb/s) that connected US supercomputer centers in 1985. But to work with colleagues across the world on petabyte data sets, scientists now demand tens of gigabits per second (Gb/s).

What many term the "Grid" offers a potential means of surmounting these obstacles to progress.¹ Built on the Internet and the World Wide Web, the Grid is a new class of infrastructure. By providing scalable, secure, high-performance mechanisms for discovering and negotiating access to remote resources, the Grid promises to make it possible for scientific collaborations to share resources on an unprecedented scale, and for geographically distributed groups to work together in ways that were previously impossible.²⁻⁴

The concept of sharing distributed resources is not new. In 1965, MIT's Fernando Corbató and the other designers of the Multics operating system envisioned a computer facility operating "like a power company or water company."⁵ And in their 1968 article "The Computer as a Communications Device," J. C. R. Licklider and Robert W. Taylor anticipated Grid-like scenarios.⁶ Since the late 1960s, much work has been devoted to developing distributed systems, but with mixed success.

IAN FOSTER is a senior scientist and associate division director in Argonne National Laboratory's mathematics and computer science division, a professor in the University of Chicago's department of computer science, and a senior fellow in the Argonne-Chicago Computation Institute.

Now, however, a combination of technology trends and research advances makes it feasible to realize the Grid vision—to put in place a new international scientific infrastructure with tools that, together, can meet the challenging demands of 21st-century science. Indeed, major science communities now accept that Grid technology is important for their future. Numerous government-funded R&D projects are variously developing core technologies, deploying production Grids, and applying Grid technologies to challenging applications. (For a list of major Grid projects, see <http://www.mcs.anl.gov/~foster/grid-projects>.)

Technology trends

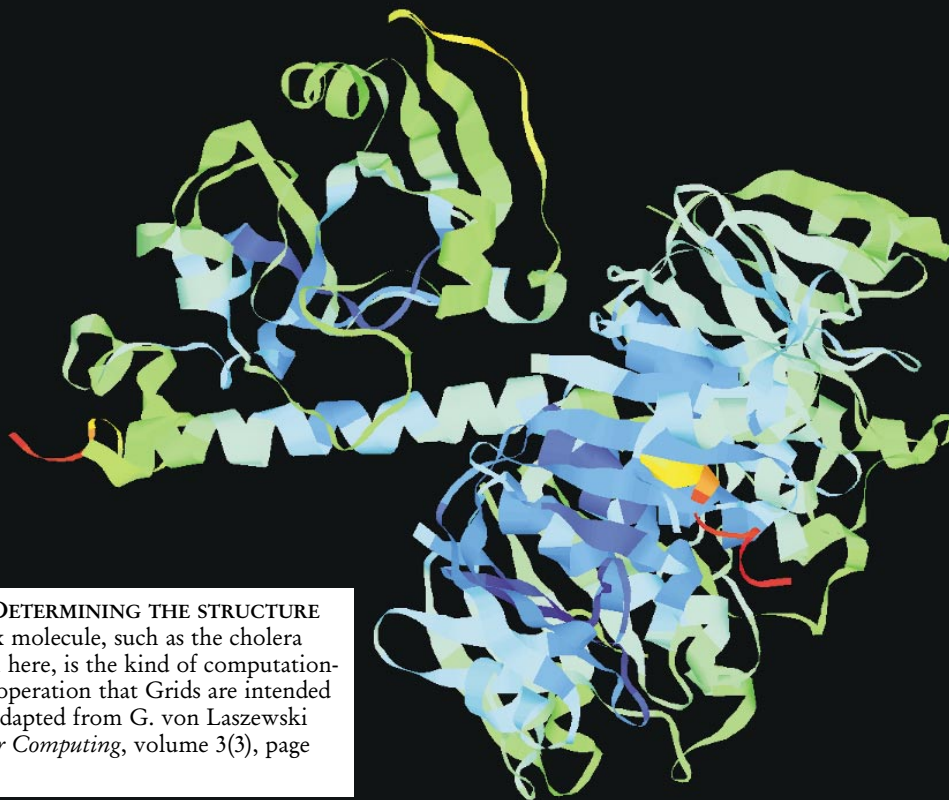
A useful metric for the rate of technological change is the average period during which speed or capacity doubles or, more or less equivalently, halves in price. For storage, networks, and computing power, these periods are around 12, 9, and 18 months, respectively. The different time constants associated with these three exponentials have significant implications.

The annual doubling of data storage capacity, as measured in bits per unit area, has already reduced the cost of a terabyte (10^{12} bytes) disk farm to less than \$10 000. Anticipating that the trend will continue, the designers of major physics experiments are planning petabyte data archives. Scientists who create sequences of high-resolution simulations are also planning petabyte archives.

Such large data volumes demand more from our analysis capabilities. Dramatic improvements in microprocessor performance mean that the lowly desktop or laptop is now a powerful computational engine. Nevertheless, computer power is falling behind storage. By doubling "only" every 18 months or so, computer power takes five years to increase by a single order of magnitude. Assembling the computational resources needed for large-scale analysis at a single location is becoming infeasible.

The solution to these problems lies in dramatic changes taking place in networking. Spurred by such innovations as doping, which boosts the performance of optoelectronic devices, and by the demands of the Internet economy,⁷ the performance of wide area networks doubles every nine months or so; every five years it increases by two orders of magnitude. The NSFnet network, which connects the National Science Foundation supercomputer centers in the US, exemplifies this trend. In 1985, NSFnet's backbone operated at a then-unprecedented 56 Kb/s. This year, the centers will be connected by the 40 Gb/s TeraGrid network (<http://www.teragrid.org>)—an

FIGURE 1. DETERMINING THE STRUCTURE of a complex molecule, such as the cholera toxin shown here, is the kind of computationally intense operation that Grids are intended to tackle. (Adapted from G. von Laszewski et al., *Cluster Computing*, volume 3(3), page 187, 2000.)



improvement of six orders of magnitude in 17 years.

The doubling of network performance relative to computer speed every 18 months has already changed how we think about and undertake collaboration. If, as expected, networks outpace computers at this rate, communication becomes essentially free. To exploit this bandwidth bounty, we must imagine new ways of working that are communication intensive, such as pooling computational resources, streaming large amounts of data from databases or instruments to remote computers, linking sensors with each other and with computers and archives, and connecting people, computing, and storage in collaborative environments that avoid the need for costly travel.⁸

If communication is unlimited and free, then we are not restricted to using local resources to solve problems. When running a colleague's simulation code, I do not need to install the code locally. Instead, I can run it remotely on my colleague's computer. When applying the code to datasets maintained at other locations, I do not need to get copies of those datasets myself (not so long ago, I would have requested tapes). Instead, I can have the remote code access those datasets directly. If I wish to repeat the analysis many hundreds of times on different datasets, I can call on the collective computing power of my research collaboration or buy the power from a provider. And when I obtain interesting results, my geographically dispersed colleagues and I can look at and discuss large output datasets by using sophisticated collaboration and visualization tools.

Although these scenarios vary considerably in their complexity, they share a common thread. In each case, I use remote resources to do things that I cannot do easily at home. High-speed networks are often necessary for such remote resource use, but they are far from sufficient. Remote resources are typically owned by others, exist within different administrative domains, run different

software, and are subject to different security and access control policies.

Actually using remote resources involves several steps. First, I must discover that they exist. Next, I must negotiate access to them (to be practical, this step cannot involve using the telephone!). Then, I have to configure my hardware and software to use the resources effectively. And I must do all these things without compromising my own security or the security of the remote resources that I make use of, some of which I may have to pay for.

Implementing these steps requires uniform mechanisms for such critical tasks as creating and managing services on remote computers, supporting single sign-on to distributed resources, transferring large datasets at high speeds, forming large distributed virtual communities, and maintaining information about the existence, state, and usage policies of community resources.

Today's Internet and Web technologies address basic communication requirements, but not the tasks just outlined. Providing the infrastructure and tools that make large-scale, secure resource sharing possible and straightforward is the Grid's *raison d'être*.

Infrastructure and tools

An infrastructure is a technology that we can take for granted when performing our activities. The road system enables us to travel by car; the international banking system allows us to transfer funds across borders; and the Internet allows us to communicate with virtually any electronic device.

To be useful, an infrastructure technology must be broadly deployed, which means, in turn, that it must be simple, extraordinarily valuable, or both. A good example is the set of protocols that must be implemented within a device to allow Internet access. The set is so small that people have

constructed matchbox-sized Web servers. A Grid infrastructure needs to provide more functionality than the Internet on which it rests, but it must also remain simple. And of course, the need remains for supporting the resources that power the Grid, such as high-speed data movement, caching of large datasets, and on-demand access to computing.

Tools make use of infrastructure services. Internet and Web tools include browsers for accessing remote Web sites, e-mail programs for handling electronic messages, and search engines for locating Web pages. Grid tools are concerned with resource discovery, data management, scheduling of computation, security, and so forth.

But the Grid goes beyond sharing and distributing data and computing resources. For the scientist, the Grid offers new and more powerful ways of working, as the following examples illustrate:

▷ **Science portals.** We are accustomed to climbing a steep learning curve when installing and using a new software package. Science portals make advanced problem-solving methods easier to use by invoking sophisticated packages remotely from Web browsers or other simple, easily downloaded “thin clients.” The packages themselves can also run remotely on suitable computers within a Grid. Such portals are currently being developed in biology, fusion, computational chemistry, and other disciplines.

▷ **Distributed computing.** High-speed workstations and networks can yoke together an organization’s PCs to form a substantial computational resource. Entropia Inc’s FightAIDSAtHome system harnesses more than 30 000 computers to analyze AIDS drug candidates. And in 2001, mathematicians across the US and Italy pooled their computational resources to solve a particular instance, dubbed “Nug30,” of an optimization problem. For a week, the collaboration brought an average of 630—and a maximum of 1006—computers to bear on Nug30, delivering a total of 42 000 CPU-days. Future improvements in network performance and Grid technologies will increase the range of problems that aggregated computing resources can tackle.

▷ **Large-scale data analysis.** Many interesting scientific problems require the analysis of large amounts of data. For such problems, harnessing distributed computing and storage resources is clearly of great value. Furthermore, the natural parallelism inherent in many data analysis procedures makes it feasible to use distributed resources efficiently. For example, the analysis of the many petabytes of data to be produced by the LHC and other future high-energy physics experiments will require

Box 1. Historical Origins

Grid concepts date to the earliest days of computing, but the genesis of much current Grid R&D lies in the pioneering work conducted on early experimental high-speed networks, such as the gigabit testbeds that were established in the US in the early 1990s.¹¹

One of these testbeds was the CASA network, which connected four laboratories in California and New Mexico. Using CASA, Caltech’s Paul Messina and his colleagues developed and demonstrated applications that coupled massively parallel and vector supercomputers for computational chemistry, climate modeling, and other sciences. Another testbed, Blanca, connected sites in the Midwest. Charlie Catlett of the National Center for Supercomputing Applications and his colleagues used Blanca to build multimedia digital libraries and demonstrated the potential of remote visualization. Two other testbeds investigated remote instrumentation. The gigabit testbeds were also used for experiments with wide area communication libraries and high-bandwidth communication protocols. Similar testbeds were created in Germany and elsewhere.

Within the US at least, the event that moved Grid concepts out of the network laboratory and into the consciousness of ordinary scientists was the I-WAY experiment.¹² Led by Tom DeFanti of the University of Illinois at Chicago and Rick Stevens of Argonne National Laboratory, this ambitious effort linked 11 experimental networks to create, for a week in November 1995, a national high-speed network infrastructure that connected resources at 17 sites across the US and Canada. Some 60 application demonstrations, spanning the gamut from distributed computing to virtual reality collaboration, showed the potential of high-speed networks. The I-WAY also saw the first attempt to construct a unified software infrastructure for such systems, the I-Soft system. Developed by the author and others, I-Soft provided unified scheduling, single sign-on, and other services that allowed the I-WAY to be treated, in some important respects, as an integrated infrastructure.

the marshalling of tens of thousands of processors and hundreds of terabytes of disk space for holding intermediate results. For various technical and political reasons, assembling these resources at a single location appears impractical. Yet the collective institutional and national resources of the hundreds of institutions participating in those experiments can provide these resources. These communities can, furthermore, share more than just computers and storage. They can also share analysis procedures and computational results.

▷ **Computer-in-the-loop instrumentation.** Scientific instruments such as telescopes, synchrotrons, and electron microscopes generate raw data streams that are archived for subsequent batch processing. But quasi-real-time analysis can greatly enhance an instrument’s capabilities. For example, consider an astronomer studying solar flares with a radio telescope array. The deconvolution and analysis algorithms used to process the data and detect flares are computationally demanding. Running the algorithms contin-

uously would be inefficient for studying flares that are brief and sporadic. But if the astronomer could call on substantial computing resources (and sophisticated software) in an on-demand fashion, he or she could use automated detection techniques to zoom in on solar flares as they occurred.

▷ **Collaborative work.** Researchers often want to aggregate not only data and computing power, but also human expertise. Collaborative problem formulation, data analysis, and the like are important Grid applications. For example, an astrophysicist who has performed a large, multiterabyte simulation might want colleagues around the world to visualize the results in the same way and at the same time so that the group can discuss the results in real time.

Real Grid applications will frequently contain aspects of several of these—and other—scenarios. For example, our radio astronomer might also want to look for similar events in an international archive, discuss results with colleagues during a run, and invoke distributed computing runs to evaluate alternative algorithms.

Grid architecture

Close to a decade of focused R&D and experimentation has produced considerable consensus on the requirements and architecture of Grid technology (see box 1 above for the early history of the Grid). Standard protocols, which define

Box 2. The Globus Toolkit

The Globus Toolkit (<http://www.globus.org>) is a community-based, open-architecture, open-source set of services and software libraries that supports Grids and Grid applications. The Toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications.

For each component, the Toolkit both defines protocols and application programming interfaces (APIs) and provides open-source reference implementations in C and (for client-side APIs) Java. A tremendous variety of higher-level services, tools, and applications have been implemented in terms of these basic components. Some of these services and tools are distributed as part of the Toolkit, while others are available from other sources. The NSF-funded GRIDS Center (<http://www.grids-center.org>) maintains a repository of components.

Globus Project and Globus Toolkit are trademarks of the University of Chicago and University of Southern California.

the content and sequence of message exchanges used to request remote operations, have emerged as an important and essential means of achieving the interoperability that Grid systems depend on. Also essential are standard application programming interfaces (APIs), which define standard interfaces to code libraries and facilitate the construction of Grid components by allowing code components to be reused.

As figure 2 shows schematically, protocols and APIs can be categorized according to the role they play in a Grid system. At the lowest level, the fabric, we have the physical devices or resources that Grid users want to share and access, including computers, storage systems, catalogs, networks, and various forms of sensors.

Above the fabric are the connectivity and resource layers. The protocols in these layers must be implemented everywhere and, therefore, must be relatively small in number. The connectivity layer contains the core communication and authentication protocols required for Grid-specific network transactions. Communication protocols enable the exchange of data between resources, whereas authentication protocols build on communication services to provide cryptographically secure mechanisms for verifying the identity of users and resources.

The resource layer contains protocols that exploit communication and authentication protocols to enable the secure initiation, monitoring, and control of resource-sharing operations. Running the same program on different computer systems depends on resource-layer protocols. The Globus Toolkit (which is described in box 2 above) is a commonly used source of connectivity and resource protocols and APIs.

The collective layer contains protocols, services, and APIs that implement interactions across collections of resources. Because

they combine and exploit components from the relatively narrower resource and connectivity layers, the components of the collective layer can implement a wide variety of tasks without requiring new resource-layer components. Examples of collective services include directory and brokering services for resource discovery and allocation; monitoring and diagnostic services; data replication services; and membership and policy services for keeping track of who in a community is allowed to access resources.

At the top of any Grid system are the user applications, which are constructed in terms of, and call on, the components in any other layer. For example, a high-energy physics analysis application that needs to execute several thousands of independent tasks, each taking as input some set of files containing events, might proceed by

▷ **obtaining** necessary authentication credentials (connectivity layer protocols)

▷ **querying** an information system and replica catalog to determine availability of computers, storage systems, and networks, and the location of required input files (collective services)

▷ **submitting** requests to appropriate computers, storage systems, and networks to initiate computations, move data, and so forth (resource protocols) and

▷ **monitoring** the progress of the various computations and data transfers, notifying the user when all are completed, and detecting and responding to failure conditions (resource protocols).

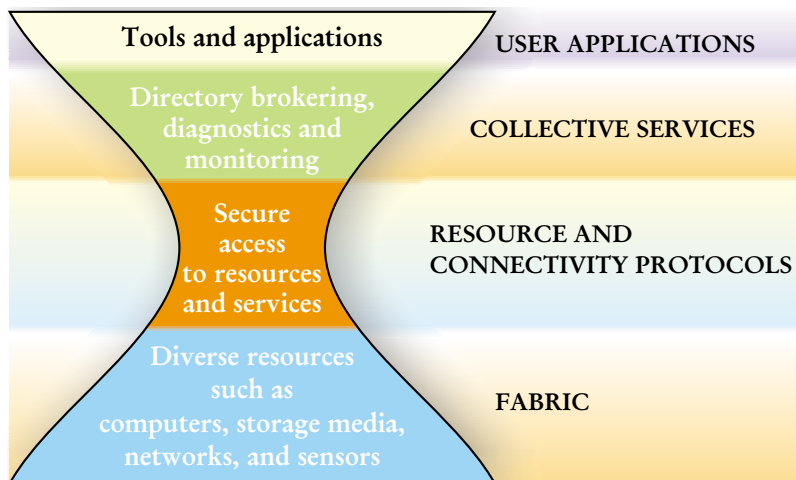
Many of these functions can be carried out by tools that automate the more complex tasks. The University of Wisconsin's Condor-G system (<http://www.cs.wisc.edu/condor>) is an example of a powerful, full-featured task broker.

Authentication, authorization, and policy

Authentication, authorization, and policy are among the most challenging issues in Grids. Traditional security technologies are concerned primarily with securing the interactions between clients and servers. In such interactions, a client (that is, a user) and a server need to mutually authenticate (that is, verify) each other's identity, while the server needs to determine whether to authorize requests issued by the client. Sophisticated technologies have been developed for performing these basic operations and for guarding against and detecting various forms of attack. We use the technologies whenever we visit e-commerce Web sites such as Amazon to buy products online.

In Grid environments, the situation is more complex. The distinction between client and server tends to disappear, because an individual resource can act as a server

FIGURE 2. GRID ARCHITECTURE can be thought of a series of layers of different widths. At the center are the resource and connectivity layers, which contain a relatively small number of key protocols and application programming interfaces that must be implemented everywhere. The surrounding layers can, in principle, contain any number of components.



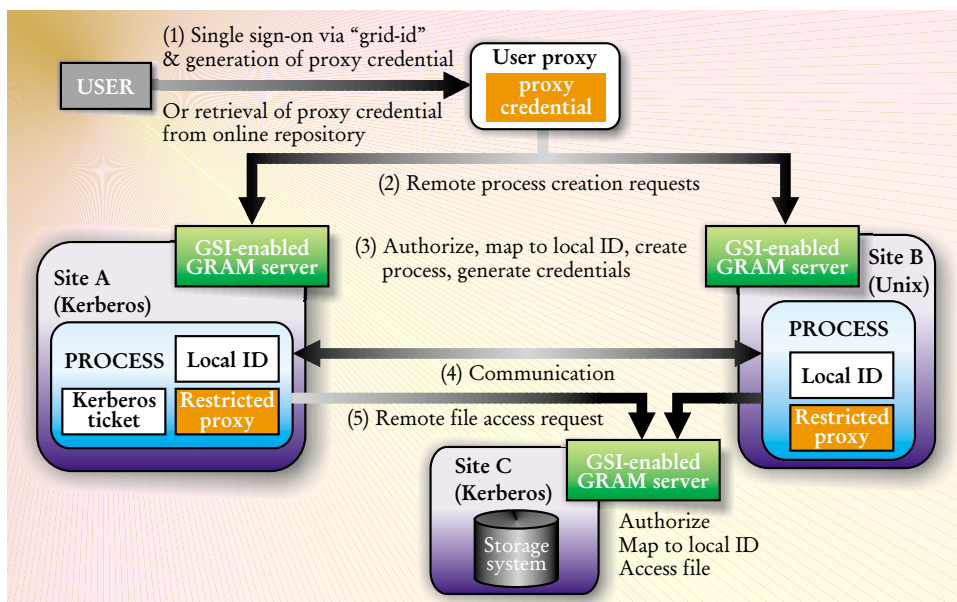


FIGURE 3. SMOOTH AND EFFICIENT authentication and authorization of requests are essential for Grid operations. Here, a user calls on the computational resources of sites A and B, which then communicate with each other and read files located at a third site, C. Each step requires authorization and authentication, from the single sign-on (or retrieval of the proxy credential) that initiates the task to the remote file access request. Mediating these requests requires the Grid Security Infrastructure (GSI), which provides a single sign-on, run-anywhere authentication service, with support for delegation of credentials to subcomputations, local control over authorization, and mapping from global to local user identities. Also required is the Grid Resource Access and Management (GRAM) protocol and service, which provides remote resource allocation and process creation, monitoring, and management services.

one moment (as it receives a request) and as a client at another (as it issues requests to other resources). For example, when I request that a simulation code be run on a colleague's computer, I am the client and the computer is a server. But a few moments later, that same code and computer act as a client, as they issue requests—on my behalf—to other computers to access input datasets and to run subsidiary computations. Managing that kind of transaction turns out to have a number of interesting requirements, such as

▷ **Single sign-on.** A single computation may entail access to many resources, but requiring a user to re-

authenticate on each occasion (by, for example, typing in a password) is impractical and generally unacceptable. Instead, a user should be able to authenticate once and then assign to the computation the right to operate on his or her behalf, typically for a specified period. This capability is achieved through the creation of a proxy credential. In figure 3, the program run by the user (the user proxy) uses a proxy credential to authenticate at two different sites. These services handle requests to create new processes.

▷ **Mapping to local security mechanisms.** Different sites may use different local security solutions, such as Kerberos and Unix as depicted in figure 3. A Grid security infrastructure needs to map to these local solutions at each site, so that local operations can proceed with appropriate privileges. In figure 3, processes execute under a local ID and, at site A, are assigned a Kerberos "ticket," a credential used by the Kerberos authentication system to keep track of requests.

▷ **Delegation.** The creation of a proxy credential is a form of delegation, an operation of fundamental importance in Grid environments.⁹ A computation that spans many resources creates subcomputations (subsidiary computations) that may themselves generate requests to other resources and services, perhaps creating additional subcomputations, and so on. In figure 3, the two subcomputations created at sites A and B both communicate with each other and access files at site C. Authentication operations—and hence further delegated credentials—are involved at each stage, as resources determine whether to grant requests and computations determine whether resources are trustworthy. The further

Box 3. Commercial Grids and the Open Grid Services Architecture

Grid concepts are becoming increasingly relevant to commercial information technology (IT). With the rise of e-business and IT outsourcing, large-scale "enterprise" applications no longer run exclusively within the friendly confines of a central computing facility. Instead, they must operate on heterogeneous collections of resources that may span multiple administrative units within a company, as well as various external networks. Delivering high-quality service within dynamic virtual organizations is just as important in business as it is in science and engineering.

One consequence of this convergence is a growing interest in the integration of Grid technologies with previously distinct commercial technologies, which tend to be based on so-called Web services. Despite the name, Web services are not particularly concerned with Web sites, browsers, or protocols, but rather with standards for defining interfaces to, and communicating with, remote processes ("services"). Thus, for example, a distributed astronomical data system might be constructed as a

set of Web services concerned variously with retrieving, processing, and visualizing data. By requiring input, such as a customer's address, in a certain format, Web services end up setting standards for remote services on the Web. Several major industrial distributed computing technologies, such as the Microsoft® .NET, IBM Corp's WebSphere, and Sun's Java™ 2 Enterprise Edition, are based on Web services.¹³

To effect the integration of Grid technologies and Web services, the Globus Project and IBM's Open Service Architecture group have proposed the Open Grid Services Architecture.¹⁴ In this blueprint, the two technologies are combined to define, among other things, standard behaviors and interfaces for what could be termed a Grid service: a Web service that can be created dynamically and that supports security, lifetime management, manageability, and other functions required in Grid scenarios. These features are being incorporated into the Globus Toolkit, and will likely also appear in commercial products.

these delegated credentials are disseminated, the greater the risk that they will be acquired and misused by an adversary. These delegation operations and the credentials that enable them must be carefully managed.

▷ Community authorization and policy.

In a large community, the policies that govern who can use which resources for what purpose cannot be based directly on individual identity. It is infeasible for each resource to keep track of community membership and privileges. Instead, resources (and users) need to be able to express policies in terms of other criteria, such as group membership, which can be identified with a cryptographic credential issued by a trusted third party. In the scenario depicted in figure 3, the file server at site C must know explicitly whether the user is allowed to access a particular file. A community authorization system allows this policy decision to be delegated to a community representative.

Current status and future directions

As the Grid matures, standard technologies are emerging for basic Grid operations. In particular, the community-based, open-source Globus Toolkit (see box 2) is being applied by most major Grid projects. The business world has also begun to investigate Grid applications (see box 3 on page 46). By late 2001, 12 companies had announced support for the Globus Toolkit.

Progress has also been made on organizational fronts. With more than 1000 people on its mailing lists, the Global Grid Forum (<http://www.gridforum.org>) is a significant force for setting standards and community development. Its thrice-yearly meetings attract hundreds of attendees from some 200 organizations. The International Virtual Data Grid Laboratory is being established as an international Grid system (figure 4).

It is commonly observed that people overestimate the short-term impact of change but underestimate long-term effects.¹⁰ It will surely take longer than some expect before Grid concepts and technologies transform the practice of science, engineering, and business, but the combination of exponential technology trends and R&D advances noted in this article are real and will ultimately have dramatic impacts.

In a future in which computing, storage, and software are no longer objects that we possess, but utilities to which we subscribe, the most successful scientific communities are likely to be those that succeed in assembling and making effective use of appropriate Grid infrastructures and thus accelerating the development and adoption of new problem solving-methods within their discipline.

I am grateful to David Abramson, Paul Avery, Fabrizio Gagliardi, Tony Hey, Satoshi Matsuoka, and Harvey Newman for their comments on an early draft of this article. My

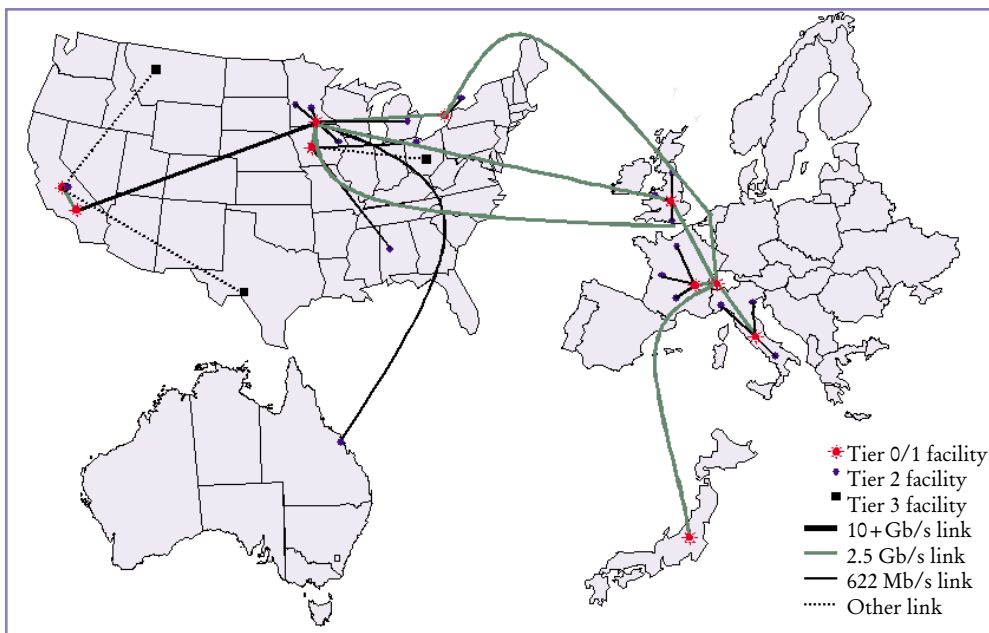


FIGURE 4. THE INTERNATIONAL VIRTUAL DATA GRID LABORATORY (iVDGL) (<http://www.ivdgl.org>) is being established to support both Grid research and production computing. The figure shows the approximate distribution of sites and networks planned for the initial rollout. (The actual sites could change by the time iVDGL becomes operational.) Major international projects, including EU DataGrid, Grid Physics Network, and Particle Physics Data Grid, are collaborating on the establishment of iVDGL.

research is supported, in part, by grants from the US Department of Energy, NSF, the Defense Advanced Research Projects Agency, NASA, and Microsoft.

References

1. I. Foster, C. Kesselman, eds., *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, San Francisco, Calif. (1999).
2. I. Foster, C. Kesselman, S. Tuecke, *Int. J. High Perform. Comput. Appl.* **15**(3), 200 (2001). Also available at <http://www.globus.org/research/papers/anatomy.pdf>.
3. *National Collaboratories: Applying Information Technology for Scientific Research*, National Academy Press, Washington, D.C. (1993). Also available at <http://www.nap.edu/books/0309048486/html>.
4. S. Teasley, S. Wolinsky, *Science* **292**, 2254 (2001).
5. V. A. Vyssotsky, F. J. Corbató, R. M. Graham, in *Fall Joint Computer Conference*, AFIPS Conf. Proc. **27**, 203 (1965). Available at <http://www.multicians.org/fjcc3.html>.
6. J. C. R. Licklider, R. W. Taylor, *Sci. Technol.*, April (1968). Also available at <http://memex.org/licklider.pdf>.
7. B. M. Leiner et al., *A Brief History of the Internet*, Internet Society, Reston, Va. (2000). Available at <http://www.isoc.org/internet-history/brief.html>.
8. L. Kleinrock, *IEEE Commun. Mag.* **30**(4), 36 (1992).
9. M. Gasser, E. McDermott, in *Proc. 1990 IEEE Symposium on Research in Security and Privacy*, IEEE Press, Piscataway, N.J. (1990), p. 20.
10. J. C. R. Licklider, *Libraries of the Future*, MIT Press, Cambridge, Mass. (1965).
11. C. Catlett, *IEEE Commun. Mag.*, April 1992, p. 42.
12. T. DeFanti et al., *Int. J. Supercomput. Appl.* **10**(2), 123 (1996).
13. S. Graham et al., *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI*, Sams Publishing, Indianapolis, Ind. (2001).
14. I. Foster et al., *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, Argonne National Laboratory, Argonne, Ill. (2002). Available at <http://www.globus.org/research/papers/physiology.pdf>. ■