



Country/region [select]

Terms of use

All of dW

Search

Home Products Services & solutions Support & downloads My account

develop

developerWorks > Grid computing >

# New to Grid computing

- developerWorks
- DB2
- eServer
- Lotus
- Rational
- Tivoli
- WebSphere
- Autonomic computing**
- Grid computing**
  - New to Grid computing
  - Downloads & products
  - Technical library
  - Training
  - Special offers
  - Events
- Java™ technology**
- Linux**
- Open source**
- Power Architecture™**
- SOA and Web services**
- Web architecture**
- Wireless technology**
- XML**
- Feedback**

- ↓ [What is grid computing?](#)
- ↓ [Why is grid computing important?](#)
- ↓ [What can I do with grid computing?](#)
- ↓ [What are the key components to grid computing?](#)
- ↓ [What standards are associated with grid computing?](#)
- ↓ [Can I build a grid today?](#)
- ↓ [How do I enable my applications for grid?](#)
- ↓ [What IBM tools are available for grid computing?](#)

Grid computing is a critical shift in thinking about how to maximize the value of computing resources. The technology is still fairly nascent, but here at the developerWorks Grid computing zone, we're publishing a steady stream of new articles, tutorials, resources, and tools to bring developers up to speed on this important, cutting-edge technology.

Many visitors interested in grid computing are asking some very basic questions:

- Where do we start?
- What do we do with all of this stuff?
- How do the pieces fit together?
- What comes next?

This page is your guide to start learning about the exciting benefits that grid computing can offer. Here, we highlight the basics of grid computing in their proper context, and we tie together relevant developerWorks articles, tutorials and tips, IBM learning services education programs, workshops, and IBM products for your further investigation. We place information about grid computing into an intuitive framework, tying the pieces together and highlighting the important details.

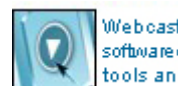
If it appears that the grid story isn't quite finished yet, then you're right -- grid technology is evolving rapidly. Standards, frameworks, implementations, and applications are changing on a constant basis. The state of grid computing today might remind you of the early days of the Web, or even of the emergence of XML and Web services, where things began slowly. But much like those technology areas, once solid standards and tools appear and coalesce, we predict there will be tremendous interest and growth in grid computing. We offer this guide so developers can get in on the ground floor.

## What is grid computing?

Because it is an emerging technology, grid computing can mean different things to different people, but here is a simple, serviceable definition for the concept of grid computing -- *grid computing allows you to unite pools of servers, storage systems, and networks into a single large system so you can deliver the power of multiple-systems resources to a single user point for a specific purpose.* To a user, datafile, or an application, the

Page options  
• [e-mail this page](#)

More resource  
• [IBM Initiative Emerging Tect Developers](#)  
• [Global Grid Fo](#)



system appears to be a single, enormous virtual computing system.

Grid computing is the next logical step in distributed networking. Just as the Internet allows users to share ideas and files as the seeds of projects, grid computing lets us share the resources of disparate computer systems so people can actually start working on those projects. Grid computing takes the ability for computers (and their users) to communicate a step further -- with grid computing, you can reach out and use computational or storage resources on machines other than your own.

With grid computing, an organization can transform its distributed and difficult-to-manage systems into a large virtual computer that can be set loose on problems and processes too complex for a single computer to handle efficiently. The problems to be solved can involve data processing, network bandwidth, or data storage. The systems linked in a grid might be in the same room, or distributed across the globe; they might be running different operating systems on many hardware platforms; they might even be owned by different organizations. Regardless of the depth of a grid's resources, all the grid user experiences is the processing resources of a very large virtual computer.

The major purpose of a grid is to virtualize resources to solve problems; the main resources grid computing is designed to give access to include (but are not limited to):

- Computing/processing power
- Data storage/networked file systems
- Communications and bandwidth
- Application software

Since the concept of putting grids into real-world practice is still relatively new, another good way to describe a grid is to describe what it isn't. The following entities are not grids:

- Cluster
- Network-attached storage device
- Scientific instrument
- Network

Each might be an important component of a grid, but by itself, doesn't constitute a grid.

So, what does it take to make the vision of the grid computing concept a reality? It requires standard and seamless, open, general-purpose protocols and interfaces, all of which are being defined now and are similar to those that enable access to information from the Web.

Learn more about it:

- The article "[Perspectives on grid: Grid computing -- next-generation distributed computing](#)" (*developerWorks*, January 2004) details how grid complements and contrasts with other forms of distributed computing.
- The redpaper "[Fundamentals of Grid Computing](#)" provides discussion material about grid computing, its concepts, use, and architecture.
- The article "[Grid computing: Conceptual flyover for developers](#)" (*developerWorks*, May 2003) explains what a developer needs to know about grid computing, including a starter list of white papers, books, and articles.
- IBM offers a [Grid computing site](#) that has white papers, analyst reports, and success stories. It explains what grid is, why it is beneficial, and how IBM can help you incorporate the technology.
- The seminal white paper "[Anatomy of the Grid](#)" by Ian Foster, Carl Kesselman, and Steven Tuecke defines the field of grid computing and focuses on its architecture.
- The white paper "[Physiology of the Grid](#)" by Ian Foster, Carl

Kesselman, Jeffrey Nick, and Steven Tuecke explains how Grid computing can be put to work in a Web services environment.

↑

[Back to top](#)

### Why is grid computing important?

Grid computing is about getting computers to work together. Almost every organization is sitting atop enormous, unused computing capacity that is widely distributed. Mainframes are idle 40% of the time. UNIX servers are actually "serving" something less than 10% of the time. And most PCs do nothing for 95% of a typical day. Imagine an airline with 90% of its fleet on the ground, an automaker with 40% of its assembly plants idle, a hotel chain with 95% of its rooms unoccupied.

Virtualization of the computing environment -- or grid computing -- is a key component of the IBM on demand strategy. Virtualization allows organizations to:

- Use otherwise idle computer resources to accelerate business processes.
- Speed applications so that processing time decreases, driving faster time to market.
- Enable the development of new and more productive applications.
- Drive down the costs of developing new applications.
- Increase collaboration and productivity capabilities.
- Maximize the resources available to users.
- Increase the resiliency and utilization of the IT environment.

Administrators and developers benefit from grid computing because it allows them to:

- Optimize the infrastructure to balance workloads, and provide extra capacity for high-demand applications.
- Improve access to data and support collaboration across disciplines, organizations, and businesses.
- Provide a more resilient infrastructure.

Businesses benefit from grid computing because it allows them to:

- Increase productivity by providing users the resources they need on demand.
- Use existing resources more efficiently.
- Respond quickly to changing business and market demands.
- Enable collaboration among dispersed entities.
- Create virtual organizations that can share resources and data.

One of the most important issues grid computing addresses for businesses is utilization of existing resources. Companies have made significant investments in computing capacity, but much of it sits idle up to 90% of the time. Grid computing can help these businesses connect those underutilized assets, harness their collective power, and manage them like a single large computer.

Learn more about it:

- Here's a list of [case studies and success stories](#) about real-world organizations deploying grids.

↑ [Back to top](#)

### What can I do with grid computing?

The concept of grid computing sprang from the research and academic

communities, much like that of the Internet, but business has recently started to catch on to the benefits that grid computing can provide. Such as, enabling new types of financial and business models like the following examples:

- In the financial services industry, grid computing can be used to speed trade transactions, crunch huge volumes of data, and provide a more stable IT environment in a mission-critical environment that doesn't tolerate much downtime.
- Government agencies can use grids to pool, secure, and integrate vast stockpiles of data. Many civilian and military agencies need the capabilities of cross-agency collaboration, data integrity and security, and lightning-fast information access across thousands of data repositories.
- Companies involved in the life sciences (such as those that do genome research and pharmaceutical development) can use parallel and grid computing to process, cleanse, cross-tabulate, and compare massive amounts of data. Faster processing means getting to market faster, and in those industries, a slight edge can be the deciding factor.

Not only *can* these new grid-oriented business models be implemented, some already have (as you can read about in the following examples).

Learn more about it:

- Examine the [business case](#) for grid computing, as well as ways to use grid computing in [various industries](#).
- Look here to find out [why grid is good for business](#).
- Grid computing is a part of an [on demand business](#).
- See [case studies and success stories](#) about real-world organizations deploying grids.

↑ [Back to top](#)

### What are the key components to grid computing?

There are six major components to grid computing:

- Security
- User interface
- Workload management
- Scheduler
- Data management
- Resource management

Let's look at each in just a bit of detail.

Computers on a grid are networked and running applications; they can also be handling sensitive or extremely valuable data, so the *security* component of grid computing is of paramount concern. This component includes elements such as encryption, authentication, and authorization.

Accessing information on the grid is also quite important, and the *user interface* component handles this task for the user. It often comes in one of two ways:

- An interface provided by an application that the user is running.
- An interface provided by the grid administrator, much like a Web portal that provides access to the applications and resources available on the grid in a single virtual space.

The portal-style interface is also important because it can be the help space for users to learn how to query the grid.

Applications that a user wants to run on a grid must be aware of the resources that are available; this is where a *workload management*

service comes in handy. An application can communicate with the workload manager to discover the available resources and their status.

A *scheduler* is needed to locate the computers on which to run an application, and to assign the jobs required. This can be as simple as taking the next available resource, but often this task involves prioritizing job queues, managing the load, finding workarounds when encountering reserved resources, and monitoring progress.

If an application is running on a system that doesn't hold the data the application needs, a secure, reliable *data management* facility takes care of moving that data to the right place across various machines, encountering various protocols.

To handle such core tasks as launching jobs with specific resources, monitoring the status of those jobs, and retrieving results, a *resource management* facility is necessary.

It's important to remember that grid computing doesn't operate in a vacuum -- just the opposite. It potentially involves every protocol and computer technology in operation today. With that in mind, we've provided links to other technologies and standards that you might need to understand to fully appreciate the scope of grid computing's power.

Learn more about it:

- The article "[Grid computing: What are the key components?](#)" (*developerWorks*, June 2003) offers a high-level overview of grid computing concepts and the components involved, plus it discusses the Globus Project.
- Learn the basics of Service-Oriented Architectures (SOAs, includes Web and grid services) in the [New to SOA and web services page](#).
- "[A visual tour of Open Grid Services Architecture](#)" (*developerWorks*, October 2003) offers a concise journey through grid components and infrastructure.
- And for an introduction to these complementary technologies and specifications:
  - [XML and XML Schema](#).
  - [SOAP](#).
  - [WSDL](#).
  - [UDDI](#).
  - [Web services](#) and its [standards](#).

[↑ Back to top](#)

### What standards are associated with grid computing?

To better understand the evolving standards for grid computing, you need to understand also how the grid architecture is defined. To do that, allow us to give you a bit of information about the definition of the architecture from the Open Grid Services Architecture (OGSA), developed by the members of the Global Grid Forum (GGF).

**The architecture.** OGSA defines what grid services are, and the overall structure and services to be provided in grid environments. Building on existing Web services standards, the OGSA defines a grid service as a Web service that conforms to a particular set of conventions. For example, grid services are defined in terms of standard WSDL (Web Services Definition Language) with minor extensions.

Why is this important? Because it gives us a common and open-standards-based set of techniques to access various grid services using existing standards, such as SOAP, XML, and WS-Security. With this base, we can add and integrate additional services (such as life cycle management) in a seamless manner. It provides a standard method to

find, identify, and utilize new grid services as they become available.

And as an added benefit, OGSA will provide for interoperability between grids that might have been built using different underlying tools.

**The specifications.** Grid specifications are evolving. Working groups in organizations like the Global Grid Forum and OASIS are busy defining an array of grid standards in areas like:

- Applications and programming models
- Architecture
- Data management
- Security
- Performance
- Scheduling and resource management

Open Grid Services Infrastructure (OGSI) is a formal specification of the concepts described by the OGSA. OGSI 1.0 specifies a set of service primitives that define a nucleus of behavior common to all grid services.

The Web Services Resource Framework (WSRF) is an evolution of OGSI 1.0. Its goal is to evolve the grid architecture in a way that's more clearly aligned with the general evolution of Web services. Instead of defining a new type of grid service, these specifications will allow the services specified in the OGSA to be based completely on standard Web services.

How much do you need to know about the evolving grid standards? It depends. IBM and other industry leaders plus researchers and representatives from many grid software vendors are actively involved in the work to define the grid standards. Are you a corporate software developer? If so, then you'll use the grid tools and products that will be based on the new standards as they unfold. You'll want to know about the standards and be generally aware of the work that's going on.

Learn more about it:

- For a high-level look at applying standards in grid computing, try the article "[Grid computing: Moving to a standardized platform](#)" (*developerWorks*, August 2003).
- This [Grid watch](#) column delivers a developer's point of view of the work going on in the standards organizations like the [Global Grid Forum](#).
- Take "[A visual tour of Open Grid Services Architecture](#)" (*developerWorks*, October 2003) and dissect the components of OGSA and learn their significance.
- The [developerWorks SOA and Web services zone](#) offers a complete roundup of articles, tutorials, news, and educational materials to get developers up to speed on Service-Oriented Architecture and Web services.
- The final [OGSI 1.0 specification](#) (PDF) and a very detailed [primer draft](#) on OGSI (Word DOC) are available at GridForge.
- Read about the proposed "[Web Services Resource Framework](#)".
- The paper "[The Physiology of the Grid](#)" (PDF) from Globus describes how grid mechanisms can implement a Service-Oriented architecture, explains how grid functionality can be incorporated into a Web services framework, and illustrates how the architecture can be applied within commercial computing as a basis for distributed system integration, both within and across organizational domains.
- "[Merging grids and Web services](#)" (*developerWorks*, September 2003) explains how the two systems are compatible and describes the benefits of using Web services in grid applications.
- The article "[Grid computing -- moving to a standardized platform](#)" (*developerWorks*, August 2003) provides an overview of how to apply standards when designing a grid.

↑ [Back to top](#)

### Can I build a grid today?

Sure. You can use both open source and vendors' proprietary tools and products to build a grid right this minute. Over time, as the grid standards solidify, you can expect vendors to enable their tools to comply with the new standards, making it easier for you to combine components that will work together.

What technologies are fundamental for building grids? Services are essential in grid computing. The services include:

- Data queries
- Data management
- Processor requests
- Workload balancing
- Job scheduling
- Bandwidth allocation

These services are called grid services. Some computers host grid services, and other computers run applications that contract grid services as clients. Grid services are essentially Web services with additional functionality.

Web services -- groups of application functions that can be invoked over a network -- allow applications to communicate with each other regardless of the platforms or programming languages involved.

To build a grid, you need tools. Grid tools fall into these general categories:

- *Infrastructure* components include filesystems, schedulers and resource managers, messaging systems, security applications, certificate authorities, and file-transfer mechanisms such as GridFTP.
- Systems on a grid must be able to discover what services are available to them -- they must be able to define (and monitor) a grid's topology in order to share and collaborate. To do this, there are grid *directory services* implementations that are based on such existing successful models as LDAP, DNS, network-management protocols, and indexing services.
- One of the main benefits of a grid is the ability to maximize efficiency; this is done through *schedulers and load balancers*. Schedulers ensure that jobs are completed in some order (such as priority, deadline, urgency, and so forth), and load balancers distribute tasks and data management across systems to decrease the chance of bottlenecks.
- *Developer tools* for grid developers focus on different niches (file transfer, communications, environment control) and range from utilities to full-blown APIs.
- *Security* in a grid environment can mean authentication and authorization -- controlling who or what can access a grid's resources -- but it can also mean such crucial issues as message integrity and confidentiality.

To build a grid today, a good place to start is to download the **Globus Toolkit 3.x** (GT3). GT3 is the first full-scale implementation of the OGSI standard. The toolkit was developed by the Globus Project, a research and development project that focuses on enabling the application of grid concepts to scientific and engineering computing. The toolkit is a set of services and software libraries designed to support grids and grid applications. The GT3 includes software for:

- Security
- Information infrastructure
- Resource management
- Data management

- Communication
- Fault detection
- Portability

Also available are Commodity Grid Kits (CoG) that provide access to grid services through a particular framework, including Java, Python, and Perl.

The IBM Grid Toolbox V3 for Multiplatforms, available free, is a set of installable packages that includes the Globus Toolkit with additional documentation and custom installation scripts written for IBM eServer hardware running AIX and Linux. IBM LoadLeveler (AIX 5.1) enablement as an alternative job manager is provided.

Learn more about it:

- Read an overview about the IBM Grid Toolbox V3 for Multiplatforms in the article "[Develop your grid service with the IBM Grid Toolbox](#)" (*developerWorks*, May 2004).
- The redbook "[Grid Services Programming and Application Enablement](#)" presents concrete programming example using the Globus Toolkit and IBM Grid Toolbox .
- "[Writing secure grid services using Globus Toolkit 3.0](#)" (*developerWorks*, October 2003) explains security terminology and mechanisms that come with the Globus Toolkit 3.0 and offers sample code to demonstrate how to implement message level security in grid services.
- You can pick up the [Java, Python, and Perl CoG kits](#) from Globus that let you access grid services through the three technologies.
- Read "[OptimalGrid -- autonomic computing on the Grid](#)" (*developerWorks*, June 2003) for information on the research prototype middleware designed to simplify creating and managing large-scale, connected, parallel grid applications.
- Download the [IBM Emerging Technologies Toolkit](#) from alphaWorks.
- Peruse [our library of grid articles](#).

↑ [Back to top](#)

### How do I enable my applications for grid?

It takes some planning.

Start by considering the basic structure of your grid and the services it provides. You have to understand how the infrastructure components fit together, including security, resource management, information services, and data management, which can affect the application architecture, design, and deployment.

Learn more about it:

- The series "[Six strategies for grid application enablement](#)" (*developerWorks*, April 2004) provides an excellent model for applications that can exploit the grid at different levels.
- The article "[Enable existing applications for the grid](#)" (*developerWorks*, June 2004) discusses how to follow a pattern to achieve the first three strategies for platform-specific distributed applications and Web-enabled applications.
- The article "[Design an application for grid](#)" (*developerWorks*, November 2003) teaches you how to enable an existing application or build one from scratch.
- "[How grid infrastructure affects application design](#)" (*developerWorks*, July 2003) is a primer on considering both the grid structure and the type of application when crafting or adapting an application for the grid.
- The redbook "[Grid Services Programming and Application Enablement](#)" illustrates the various steps needed to develop a grid

service application.

- See all the [redbooks](#) about grid and on-demand.
- The redbook "[Enabling Applications for Grid Computing with Globus](#)" focuses on the issues and considerations for grid-enabling an application, and contains many examples of C/C++ and Java programs.
- Look into [courses](#) offered by IBM Global Services
- Learn some basics and get hands-on experience in the tutorial series "[Build a grid with Perl](#)" and in the tutorial series "[Build a grid application with Python](#)".
- The article series "[Business service grid](#)" shows how to build a grid solution using [Service Domain technology](#) available from IBM [alphaWorks](#).
- These [grid application tutorials](#) will provide hands-on practice in enabling and building grid applications.
- Get practical tips from the articles in our [technical library](#).

[↑ Back to top](#)

### What IBM tools are available for grid computing?

When it comes to IBM products, the overall strategy is to "grid-enable" them by working toward specific OGSA compliance for:

- Storage
- Server
- Infrastructure
- Database management
- Systems management
- Messaging and file systems offerings

The company also is using OGSA as the common foundation for new technologies and products in development.

Here are some of the IBM grid-oriented tools that can help you leverage the power of grid computing:

- The [IBM Grid Toolbox](#) is a set of installable packages that include the Globus Toolkit with additional documentation and custom installation scripts written for IBM eServer hardware for multiple platforms.
- The [IBM DB2 Information Integrator](#) provides a foundation for e-business on demand, enabling companies to have integrated, real-time access to diverse and distributed information.
- The [IBM Emerging Technologies Toolkit](#) from alphaWorks is a software development kit for designing, developing, and executing emerging autonomic and grid-related technologies and Web services.
- [Grid application framework for Java](#) from alphaWorks is a lightweight framework that abstracts all grid semantics from the application logic and provides a simpler programming model that lines up smoothly with common Java programming models.
- The [IBM Grid Toolbox V3 for Multiplatforms](#) incorporates the Globus Toolkit plus administrative tools, APIs, and a streamlined installation.
- [IBM Tivoli Intelligent ThinkDynamic Orchestrator](#) automates the steps to provision, configure and deploy a solution into production. It supports servers, operating systems, middleware, applications and network devices acting as firewalls, routers, switches, and load balancers.
- [IBM Tivoli Provisioning Manager](#), through workflows, automates the manual provisioning and deployment of servers, operating systems, middleware, applications, virtual environments, SAN- and NAS-based storage resources, and network devices acting as routers, switches, firewalls, and load balancers.
- [IBM TotalStorage SAN File System](#), also called StorageTank, helps

reduce the complexity of managing files within SANS.

- [IBM TotalStorage SAN Volume Controller](#) centrally manages multiple storage systems and combines the capacity from multiple disk storage systems into a single storage pool.
- [IBM Virtualization Engine Suite for Servers](#) is a set of comprehensive system technologies and services.
- Read "[OptimalGrid -- autonomic computing on the Grid](#)" (*developerWorks*, June 2003) for information on the research prototype middleware designed to simplify creating and managing large-scale, connected, parallel grid applications. It's available from IBM alphaWorks.
- This [grid tools list](#) will provide you with tools to cover the entire spectrum of crafting, installing, and managing an effective grid system.

[↑ Back to top](#)

[About IBM](#)

[Privacy](#)

[Contact](#)