**Grid computing** is a term referring to the combination of computer resources from multiple administrative domains to reach common goal. What distinguishes grid computing from conventional high performance computing systems such as cluster computing is that grids tend to be more loosely coupled, heterogeneous, and geographically dispersed. Although a grid can be dedicated to a specialized application, it is more common that a single grid will be used for a variety of different purposes. Grids are often constructed with the aid of general-purpose grid software libraries known as middleware.

Grid size can vary by a considerable amount. Grids are a form of distributed computing whereby a “super virtual computer” is composed of many networked loosely coupled computers acting together to perform very large tasks. Furthermore, “Distributed” or “grid” computing in general is a special type of parallel computing that relies on complete computers (with onboard CPUs, storage, power supplies, network interfaces, etc.) connected to a network (private, public or the Internet) by a conventional network interface, such as Ethernet. This is in contrast to the traditional notion of a supercomputer, which has many processors connected by a local high-speed computer bus.

**Overview**

Grid computing combines computer resources from multiple administrative domains to reach common goal. Grid computing (or the use of a computational grid) simultaneously applies the resources of many computers in a network to tackle a single problem, usually to solve a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. When many are brought together in this collaborative effort, they are known as Virtual Organizations (VOs). These VOs may be formed to solve a single task and may then disappear just as quickly.

One of the main strategies of grid computing is to use middleware to divide and apportion pieces of a program among several computers, sometimes up to many thousands. Grid computing involves computation in a distributed fashion, which may also involve the aggregation of large-scale cluster computing-based systems. The grid is also sometimes referred to as the replacement for the Internet, offering speeds potentially 10,000 times faster than traditional broadband. This was pushed for in a large part by the research center CERN and its need for super fast Internet to send information across the globe in order to analyse and store its massive amounts of research data.

The size of a grid may vary from small—confined to a network of computer workstations within a corporation, for example—to large, public collaborations across many companies and networks. "The notion of a confined grid may also be known as an intra-nodes cooperation whilst the notion of a larger, wider grid may thus refer to an inter-nodes cooperation."
Grids are a form of distributed computing whereby a “super virtual computer” is composed of many networked loosely coupled computers acting together to perform very large tasks. This technology has been applied to computationally intensive scientific, mathematical, and academic problems through volunteer computing, and it is used in commercial enterprises for such diverse applications as drug discovery, economic forecasting, seismic analysis, and back office data processing in support for e-commerce and Web services.

Comparison of grids and conventional supercomputers

“Distributed” or “grid” computing in general is a special type of parallel computing that relies on complete computers (with onboard CPUs, storage, power supplies, network interfaces, etc.) connected to a network (private, public or the Internet) by a conventional network interface, such as Ethernet. This is in contrast to the traditional notion of a supercomputer, which has many processors connected by a local high-speed computer bus. The primary advantage of distributed computing is that each node can be purchased as commodity hardware, which, when combined, can produce a similar computing resource as multiprocessor supercomputer, but at a lower cost. This is due to the economies of scale of producing commodity hardware, compared to the lower efficiency of designing and constructing a small number of custom supercomputers. The primary performance disadvantage is that the various processors and local storage areas do not have high-speed connections. This arrangement is thus well-suited to applications in which multiple parallel computations can take place independently, without the need to communicate intermediate results between processors. The high-end scalability of geographically dispersed grids is generally favorable, due to the low need for connectivity between nodes relative to the capacity of the public Internet.

There are also some differences in programming and deployment. It can be costly and difficult to write programs that can run in the environment of a supercomputer, which may have a custom operating system, or require the program to address concurrency issues. If a problem can be adequately parallelized, a “thin” layer of “grid” infrastructure can allow conventional, standalone programs, given a different part of the same problem, to run on multiple machines. This makes it possible to write and debug on a single conventional machine, and eliminates complications due to multiple instances of the same program running in the same shared memory and storage space at the same time.

Design considerations and variations

One feature of distributed grids is that they can be formed from computing resources belonging to multiple individuals or organizations (known as multiple administrative domains). This can facilitate commercial transactions, as in utility computing, or make it easier to assemble volunteer computing networks.
One disadvantage of this feature is that the computers which are actually performing the calculations might not be entirely trustworthy. The designers of the system must thus introduce measures to prevent malfunctions or malicious participants from producing false, misleading, or erroneous results, and from using the system as an attack vector. This often involves assigning work randomly to different nodes (presumably with different owners) and checking that at least two different nodes report the same answer for a given work unit. Discrepancies would identify malfunctioning and malicious nodes.

Due to the lack of central control over the hardware, there is no way to guarantee that nodes will not drop out of the network at random times. Some nodes (like laptops or dialup Internet customers) may also be available for computation but not network communications for unpredictable periods. These variations can be accommodated by assigning large work units (thus reducing the need for continuous network connectivity) and reassigning work units when a given node fails to report its results in expected time.

The impacts of trust and availability on performance and development difficulty can influence the choice of whether to deploy onto a dedicated computer cluster, to idle machines internal to the developing organization, or to an open external network of volunteers or contractors. In many cases, the participating nodes must trust the central system not to abuse the access that is being granted, by interfering with the operation of other programs, mangling stored information, transmitting private data, or creating new security holes. Other systems employ measures to reduce the amount of trust “client” nodes must place in the central system such as placing applications in virtual machines.

Public systems or those crossing administrative domains (including different departments in the same organization) often result in the need to run on heterogeneous systems, using different operating systems and hardware architectures. With many languages, there is a trade off between investment in software development and the number of platforms that can be supported (and thus the size of the resulting network). Cross-platform languages can reduce the need to make this trade off, though potentially at the expense of high performance on any given node (due to run-time interpretation or lack of optimization for the particular platform).

Various middleware projects have created generic infrastructure to allow diverse scientific and commercial projects to harness a particular associated grid or for the purpose of setting up new grids. BOINC is a common one for various academic projects seeking public volunteers; more are listed at the end of the article.

In fact, the middleware can be seen as a layer between the hardware and the software. On top of the middleware, a number of technical areas have to be considered, and these may or may not be middleware independent. Example areas include SLA management, Trust and Security, Virtual organization management, License Management, Portals and Data Management. These technical areas may be taken care of in a commercial solution,
though the cutting edge of each area is often found within specific research projects examining the field. [citation needed]

**Market segmentation of the grid computing market**

According to IT-Tude.com, for the segmentation of the grid computing market, two perspectives need to be considered: the provider side and the user side:

**The provider side**

The overall grid market comprises several specific markets. These are the grid middleware market, the market for grid-enabled applications, the utility computing market, and the software-as-a-service (SaaS) market.

Grid middleware is a specific software product, which enables the sharing of heterogeneous resources, and Virtual Organizations. It is installed and integrated into the existing infrastructure of the involved company or companies, and provides a special layer placed among the heterogeneous infrastructure and the specific user applications. Major grid middlewares are Globus Toolkit, gLite, and UNICORE.

Utility computing is referred to as the provision of grid computing and applications as service either as an open grid utility or as a hosting solution for one organization or a VO. Major players in the utility computing market are Sun Microsystems, IBM, and HP.

Grid-enabled applications are specific software applications that can utilize grid infrastructure. This is made possible by the use of grid middleware, as pointed out above.

Software as a service (SaaS) is “software that is owned, delivered and managed remotely by one or more providers.” (Gartner 2007) Additionally, SaaS applications are based on a single set of common code and data definitions. They are consumed in a one-to-many model, and SaaS uses a Pay As You Go (PAYG) model or a subscription model that is based on usage. Providers of SaaS do not necessarily own the computing resources themselves, which are required to run their SaaS. Therefore, SaaS providers may draw upon the utility computing market. The utility computing market provides computing resources for SaaS providers.

**The user side**

For companies on the demand or user side of the grid computing market, the different segments have significant implications for their IT deployment strategy. The IT deployment strategy as well as the type of IT investments made are relevant aspects for potential grid users and play an important role for grid adoption.

**CPU scavenging**
CPU-scavenging, cycle-scavenging, cycle stealing, or shared computing creates a “grid” from the unused resources in a network of participants (whether worldwide or internal to an organization). Typically this technique uses desktop computer instruction cycles that would otherwise be wasted at night, during lunch, or even in the scattered seconds throughout the day when the computer is waiting for user input or slow devices.

Volunteer computing projects use the CPU scavenging model almost exclusively.\[citation needed]\n
In practice, participating computers also donate some supporting amount of disk storage space, RAM, and network bandwidth, in addition to raw CPU power. Heat produced by CPU power in rooms with many computers can be used for fine heating premises.\[7\] Since nodes are likely to go "offline" from time to time, as their owners use their resources for their primary purpose, this model must be designed to handle such contingencies.

Taxation issues

The project BEinGRID has studied the legal issues involved in grid computing. In particular, the issue of tax is crucial. The first question to answer is why taxation issues are likely to be relevant in a grid environment. This is a consequence of the distributed nature of the service provided: potentially taxable income may be generated from the use in combination of servers that are located in various tax domains. The problems that tax consultants and managers have to face are thus in many cases cumbersome and novel.\[8\]

Firstly, they must find out which VAT rules are applicable to a European ICT company that is willing to provide e-services to businesses or individuals located in the same country, in another European country or outside of the EU. Also, it needs to be decided what services can be considered e-services (electronically supplied services). The solutions are based on the applicable European country law sources, namely Directive 112/2006/EC, including the amendments introduced by Directive 2008/8/EC. Secondly, regarding international income taxation, they must conclude whether a single server, node, etc. of a grid infrastructure should be considered and if it is a permanent establishment (PE) of the company. The corresponding solutions are likely to have a great impact on the concrete business of ICT undertakings, and taxation is one of the most important drivers when drafting business plans.