Chapter 1

A New Computing Model for the Internet

Just when people began to talk about the "end of the PC era," at the beginning of 2000, peer-to-peer (P2P) computing burst onto the scene.

The concept isn't new. What's new is the enthusiasm with which developers and users are embracing peer computing. Napster, a music file-sharing application, brought P2P to the forefront by attracting millions of users, saturating colleges' networks, and raising some fundamental questions about copyrights and digital distribution of content.

Exchanging files, which is what the Napster application enables, is just one way of sharing resources among networked computers. Sharing and collaborating via the Internet today can happen in many other ways as well:

- Dividing the load of performing large computations
- Collaborating in creating media or software
- Conversing online and directly
- Organizing into online communities

Peer-to-peer computing is evolving into a phenomenon whose impact will be no less than that which we experienced when the graphical-interface browser and, again, when the client-server computing model were introduced.

The excitement around P2P was a catalyst for an important gathering. Over 350 people, representing about 250 companies and organizations, got together on October 12, 2000, in a San Jose hotel, for the first Peer-to-Peer Working Group meeting. This occasion was significant on at least two counts: It was a testament to the fast pace at which the user community embraces various peer-to-peer applications. And it brought together the Corporate and the Start-Up worlds, competitors, and colleagues alike—developers from a whole host of disciplines. The computer industry started to work together on common issues related to P2P computing.

The event was not, by itself, a turning point, but it validated a level of acceptance by the Industry of the significance of P2P. There were other meetings, public and behind closed doors, which advanced P2P computing. And much of what has happened in the computer industry in the last 20-25 years laid the groundwork for widespread use of P2P computing.

This is the story of peer-to-peer computing—from past experimental projects, to the present resurgence, and its future as a computing model for businesses and consumers on the Internet.

This book shows how P2P computing can enrich our net experience in many ways. This promise can best be realized if cross-platform and cross-application interoperability, preferably through a common P2P infrastructure, comes into being.

What is Peer-to-Peer Computing?—First Take

A good starting point is to explain P2P computing in terms of the client-server architecture for network-based computing. In a client-server model, the client (the computer where the user is) makes requests of the server to which it is networked. The server, typically an unattended system in a back room, responds to, and acts on, the requests. The idea behind P2P computing is that each *peer*, i.e., each participating computer, can act both as a client and as a server *in the context of some application*.

Computers in P2P computing are not rigidly defined as either clients or servers. They can be both, to some extent,[1] and "peer" is the word that best expresses the new role and relationship of these computers.[2] So, a peer is a computer that behaves as a client in the client-server model, but also contains an additional layer of software that allows it to perform server functions. The peer computer can respond to requests from other peers. The scope of the requests and responses, and how they are executed are application-specific. Typically, there is a request for access to resources that belong to the other peer. The request may be for a file to be read or copied, information on con-

1 The extent is limited because a desktop computer may have some server functionality, but is not likely to be able to replace a large multi-processor database server, for example.

2 Deriving from the familiar client-server model some developers call this new type of end-user computer a *servent*, and others call it *clerver*.

tent and files, computations to be performed, a message or a file to be passed on to others, etc.

The client-server duality exhibited by a peer is a central feature of P2P.

The client-server model and the P2P model are not exclusive of each other. The two models can, and do, co-exist. The addition of some server functions to the client, making it a peer, does not diminish its capability to act as a full-fledged client towards servers on the net. Later you will read about the so-called hybrid model of peer computing where servers are used strictly as servers in the client-server sense.

The term *P2P computing* can refer to the model, or it can refer to the action of computing within that framework. Both facets are useful in defining P2P computing.

The P2P framework allows peers to interact with each other directly. An important characteristic of this direct interaction is that the computing environment becomes *decentralized.*[3] Think of the client-server model as a bicycle wheel where the server is at the center, the clients reside on the rim, and the spokes are the communication lines. In the P2P model, the spokes connect peers directly (see Figure 1.1). In reality, there may be connections to the center of the wheel, but not everything needs to go through the central server. As a result, the effect of P2P computing is to take computing to the *edges of the net*.

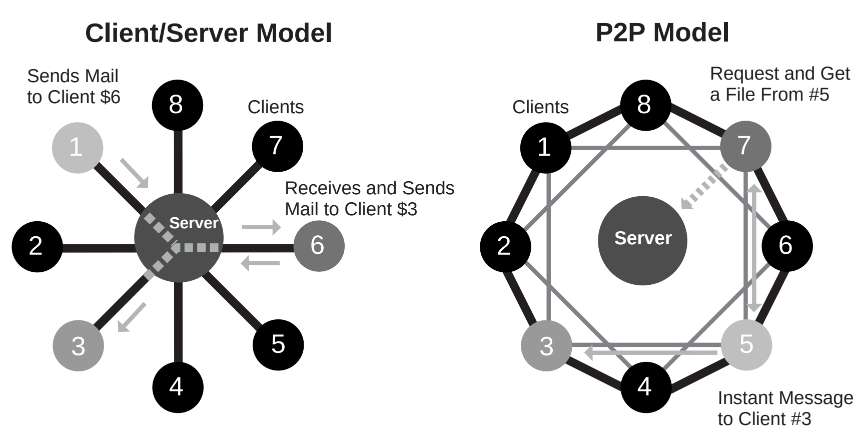


Figure 1.1 P2P Computing is Decentralized and at the Edges of the Net

[3] For a thought provoking article on decentralized systems, see Mitchel Resnick's "Beyond the Centralized Mindset" (Resnick, 1996).

When the “computing” in P2P computing is an action, it refers to what we do with the P2P framework. You will read later about many end-user applications that become possible through the P2P services. These applications include storage, computations, messaging, security, distribution, and more. What unifies these application types in the context of P2P is that they involve *sharing* of resources and some form of *collaboration*. Resource sharing and collaboration among peers are the distinguishing properties of P2P applications.

What does P2P computing mean to the user? Direct exchanges with other users, on their peer systems, liberates the users from some of the dependence on central servers. The tools for resource sharing empower users by allowing them to initiate services and provide access to resources under their own control.

From a social viewpoint, P2P computing enables users to organize in groups without the assistance of a central authority. These self-governed communities can share, collaborate, and communicate, or participate in their own private web. The emergence of dynamic online communities at the edges of the net is one of the more exciting phenomenon brought about by P2P computing.[4]

In One Word: Interoperability

Much of the interest surrounding P2P computing is based not so much on what has been demonstrated so far, but on its potential. P2P empowers users. It enables them to accomplish tasks previously done only by some remote server that someone else is managing. P2P allows you and your friends, or you and your family, to create your own virtual world of exchanges and communications. This is what P2P means for the consumer.

P2P also impacts business-to-business interactions as well as business-to-customer interactions. Collaboration and resource sharing take on new dimensions—productivity is increased and costs are cut through the use of huge amounts of resources that previously were unused. P2P computing is a formidable ally in streamlining the business supply chain, and filling the missing links for an effective e-Commerce.

The potential for benefits from P2P computing is enormous. And we, the Industry and the developer community, can seize this opportunity by enabling *interoperability*. All the uses of P2P so far work as isolated pieces. For each

[4] Other terms are also used for “online communities.” Examples are “virtual organizations” (Foster et al., 2001), or “coalitions,” or “federations,” or just “groups.”

application, the developers create their own underlying services and tools and the applications cannot "talk" to each other.

Interoperability increases the productivity of developers and allows for applications to communicate with each other, but there are other equally important benefits. Interoperability means that P2P computing can span a multitude of platforms, with different hardware architectures running various operating systems, and that different device types can cooperate in the context of P2P computing. Servers, workstations, desktop and mobile PCs, handheld and wireless devices, even sensor-like appliances can all take part in some P2P application.

There are a number of ways to achieve interoperability, but the most effective way is through a *common infrastructure.*[5] Once we have a common infrastructure, or middleware, for P2P computing, the application developers do not need to reinvent the wheel for each new application. The application will run on a variety of hardware and software platforms while communicating with other devices. And users benefit from working in a single environment where applications are integrated to perform more complex tasks or share information and exchange data smoothly.

A number of technical and societal barriers need to be overcome before a broad acceptance of P2P is a reality. A key ingredient for addressing these challenges and assuring a successful adoption of P2P computing is interoperability. In summary:

- Interoperability allows P2P applications to run on a variety of hardware and software platforms.
- Interoperability allows for exchanges between different device types in a P2P computing environment.
- Interoperability allows communication between, and integration of, P2P applications.
- Common services reduce development efforts and improve time-to-market for application developers.

The Popular View of Peer-to-Peer

A great number of computer users, and many developers, first became aware of P2P computing near the beginning of the year 2000. That was when companies like Scour (http://www.scour.com/) and Napster (http://www.napster.com/) became magnets to huge numbers of users for

[5] Other approaches to interoperability may involve interfaces and bridges among a variety of different infrastructures.

services involving direct swapping of MP3 music files. Napster was a phenomenon. Though it only went live in September 1999, by mid-2001 it had over 50 million users. It had also brought upon itself the wrath of the established record industry because music files were copied and passed on free of charge. The impact of using online MP3 file sharing to distribute music revolutionized what had been a well-established business model. This is a classic example of a "disruptive technology" (Christensen, 1997) in action. This book, however, is confined to the technical aspects of P2P computing. The most basic view of P2P computing, associated with Napster, is that it offers an efficient method for sharing, swapping, and distributing music in digital format. This view is simplistic and incomplete and does not capture the true nature and potential of P2P computing.

Scour[6] actually appeared on the scene before Napster, as early as December 1997. It specialized in sharing media files such as video and photos, in addition to MP3 files. Napster also offers other P2P-related capabilities such as Instant Messaging. You will read more about these capabilities in subsequent chapters of this book.

Other non-music P2P applications are also popular. The Freenet Project asserts that it supports free speech on the net by providing anonymity in direct exchanges among its users. Anonymity here means that it not easy to trace back the source from which a communication originated. Here is how the Freenet Project defines itself:

> Freenet is a peer-to-peer network designed to allow the distribution of information over the Internet in an efficient manner, without fear of censorship. Freenet is *completely decentralized*, meaning that there is no person, computer, or organisation in control of Freenet or essential to its operation. (http://freenet.sourceforge.net/ -- Nov. 2000)

Note the important concept of *decentralization*—this is a recurring theme when we discuss P2P architecture and applications.

You have probably heard about Gnutella's "fully-distributed information-sharing technology" (http://gnutella.wego.com/—Nov. 2000). The features that characterize the Gnutella framework are any-type file sharing, almost complete anonymity, and the resilience of the service. This last item refers to the fact that the framework of exchanges does not have a single point of failure.

The applications cited so far deal with sharing different types of files and textual exchanges among the participants. Other projects, involving a smaller

[6] The Scour technology is now owned by CenterSpan, another company engaged in distribution of digital content over the Internet.

community of users, entail another type of P2P computing. And this time the term “computing” is even more appropriate. Computers on the net can share more than copies of their files or space for messages and text. They can share their processing power by working together to process data and solve computational tasks of great complexity and magnitude. This is called “*cycle sharing*,” and I will describe some fascinating examples in chapter 11.

For now, here is an interesting example of cycle sharing: By October 2000, 2.4 million users from over 200 countries contributed time on their computers to help in a scientific experiment that uses Internet-connected computers in the Search for Extraterrestrial Intelligence (SETI). The users contribute their idle cycles while the guest application runs as a screensaver, and the aggregate performance is bigger than that of the most powerful supercomputer built to date. Each participant gets a chunk of work that takes, typically, several days to complete. The results are then sent in, by the application, to the central database. SETI is a scientific effort to determine whether there is intelligent life out in the universe. The SETI@home project (http://setiathome.ssl.berkeley.edu/) involves analyzing data collected from the billions of radio frequencies that flood the universe, looking for a pattern of signals that indicate the existence of another civilization. This is just one example of the power of P2P-like distributed processing in tackling major computational problems. For an interesting history and case studies of distributed computing for the masses, see *Collective Wisdom* (Hayes, 1998).

Is SETI@home a “true” P2P application? After all, the participants work towards a common goal, but have no communication among themselves, and the computations are controlled by a central server. However, SETI is still P2P because it involves a high degree of collaboration without direct communication between participants. This application is quite common in solving large problems via distributed processing. Peers are cooperating by sharing their resources over the net, and this is yet another manifestation of the possibilities offered by P2P computing.

Many other companies and researchers are engaged in P2P computing while this book is being written. The examples cited here are just a few that received public attention in the early stages of the P2P emergence.

To appreciate the level of activity of P2P computing and the variety of P2P applications, view, for example, the ever-changing list on the PeerProfits site (http://www.peerprofits.com/).

The applications that triggered the emerging P2P computing movement share several salient features:

- The applications span a wide range of application types.
- The applications are decentralized.

- Being decentralized adds resilience against failures and promises a steady and reliable service.
- The application developers emphasize distributed services and distributed processing.

However, each manifestation of a P2P application is self-contained and does not interact with any other P2P application.

As you read on, P2P computing will be defined and described more rigorously, with emphasis on how to take advantage of the strengths of this computing model, and how to deal with its present weaknesses.

Why and Why Now?

What are the factors that came together and allowed and accelerated the emergence of the P2P phenomenon?

First, there was the Internet. Then came the *hyperlink* that allowed easy access and viewing of linked information from within a text document. Multiple protocols drove multiple webs; Telnet, FTP, Gopher, WAIS, and HTTP were the main ones. This was the first generation of the net.

The second generation came about with the advent of the graphical-interface browser, Mosaic, specifically. The functionality built into its graphical interface made the retrieval process transparent to the user. And, through the use of the browser, all the separate webs merged into a single World Wide Web.

And now we are on the threshold of the third generation of the Internet, ushered in by the P2P technology and the computing model it represents. This potent combination will make us members of decentralized, self-organizing online communities. They will change the electronic social landscape and the ways in which e-Commerce and corporate computing are managed and conducted.

Why is P2P computing a good thing and why is it catching on? The P2P model offers several compelling advantages to individual users and large organizations. These advantages can be divided into technical benefits and social appeal.

Technically, P2P provides the opportunity to get back huge amounts of *untapped resources* that go unused without it. The resources include processing power for large scale computations and an enormous storage potential. P2P allows the elimination of the *single source* bottleneck. Data and control can be distributed and requests can be load-balanced across the net. In addition to the performance issue, this mechanism also eliminates the *single point* *of failure* hazard. Within the enterprise, IT operations may be able to replace some costly data center functions by enabling *distributed services between clients*. Storage, for data and its backup, can be placed on clients. The P2P infrastructure, by allowing access and shared space, improves *remote maintenance* capabilities.

Much of the wide appeal of P2P is due to social and psychological factors. People can create their own online communities easily and run them as they collectively choose. Others like the ability to bypass all centralized control. For better or worse, almost perfect anonymity is possible.

And why now? The concept that underlies the P2P computing model, direct exchanges and sharing of files and cycles between nodes, is not new. Why has the P2P model burst on the scene now?

Several essential developments have reached a stage that makes widespread use of P2P possible. The ubiquity of connected computers has enabled more and more anywhere, anytime access to the net and its resources. There certainly is a critical mass of computer users. The communications bandwidth, still growing fast, makes it possible to move large amounts of data and rich media content from one location to another. The computers have enough processing power and storage capacity to handle the extra services required in a P2P environment.

New technology becomes widely used after it has become viable. It needs to be easy to use by novice users and the tools it requires have to be widely available. In the case of P2P, these tools are today's desktop computers in combination with the net. The average user communicates comfortably on the web, sending and retrieving information, searching, downloading, and even trading. The P2P services are easy to install, and their user interface is simple and familiar.

Another element that accelerates adoption is the emergence of complementary technologies. Three such technologies are noteworthy. The first is *broadband communication*. A mid-February, 2001 report[7] finds some form of broadband for Internet access in 9% of connected households in the U.S., and predicts fast growth rate of subscriptions to DSL and cable modems. Broadband communication makes persistent exchanges and connectivity more palatable to many users. Secondly, the increased popularity of wireless devices adds another dimension to P2P computing and offers opportunities for new applications. Thirdly, advances in software agent technologies provide mechanisms for managing the complexity inherent in P2P computing and

[7] Cahners In-Stat Group (www.instat.com) report dated February 14, 2001.

avenues for interesting applications. The topic of complexity is noted in Chapter 12, and intelligent agents are discussed in Chapter 7.

These conditions for the introduction of P2P technologies are necessary, but not sufficient. History shows that a *trigger* is needed for a new technology to really take off. In the computer industry this is often referred to as the *killer app*. The PC was embraced in volume when the electronic spreadsheet appeared. Closer and more relevant to P2P is the transformation from the limited use of the Internet into the World Wide Web experience we know today. The trigger was the Mosaic browser. In the year before its introduction, 50 web servers were on the Internet. A year after Mosaic appeared, the Internet had some 10,000 web servers, *and* the first web-based businesses started.

The P2P computing model found its trigger for wide acceptance during the few months spanning the end of 1999 and the beginning of 2000. The trigger was Napster, followed by Gnutella. Not only did their popularity get everyone talking about P2P, it also brought to the fore other uses of P2P. Projects such as Freenet, SETI@home, and GIMPS[8] received increased visibility by the public at large. By the fourth quarter of 2000, over 100 companies and numerous research projects were engaged in P2P computing.

What is Peer-to-Peer Computing

So far, there is no agreement on a succinct definition of P2P. People also disagree on the scope of applications that may be "true" P2P applications. Do not be overly concerned with this apparent lack of unity among P2P practitioners. Because this new computing model is rapidly evolving, it is healthy and desirable *not* to be locked down by a rigid definition. The introduction of any new technology introduces surprises, and we need to accommodate them. Also, when P2P stakeholders list the important attributes of a P2P environment, their answers are reasonably consistent.

[8] GIMPS stands for the Great Internet Mersenne Prime Search. See http://www.mersenne.org/prime.htm for more information. Both SETI@home and GIMPS predate Napster.

This variety of opinions regarding the definition of P2P was evident in a meeting, the Peer-to-Peer Summit, called by O'Reilly & Associates a couple of weeks before the first P2P Working Group meeting (Oram, 2000). A few months later, in the mid-February O'Reilly Peer-to-Peer Conference, people were still evolving their personal definitions of P2P computing. You can explore some of these views by browsing the O'Reilly OpenP2P.com site (http://www.openp2p.com/). The same was true at a number of other public gatherings that have taken place recently. The participants in all these meetings were some of the leading proponents of the P2P movement: technologists, researchers, heads of P2P start-ups, and journalists. They did not tell us, collectively, what P2P is. They did point out, individually, what the impact of P2P computing might be, some of the important ingredients to its success, and a number of barriers that need to be overcome to achieve wide acceptance of P2P computing. At least some of the insights expressed at that meeting are reflected, in one form or another, in later chapters of this book.

All that said, a more formal definition sums up this book so far:

> *Peer-to-peer computing is a network-based computing model for applications where computers share resources via direct exchanges between the participating computers.*

This definition highlights two elements that are fundamental to P2P computing—the sharing of resources and direct communication between users. These two together are a common thread in describing P2P applications.

Others may prefer different definitions, perhaps focused on collaboration and moving computing to the edges of the net, or centered on the types of services that support P2P. These are all valid approaches. At this time, it seems more important to agree on the scope of P2P, and that is the discussion that follows.

You can grasp the "spirit" of P2P from Figure 1.2. It provides a visual illustration of how a file is shared in the client-server model, and how it is done using the P2P computing model.

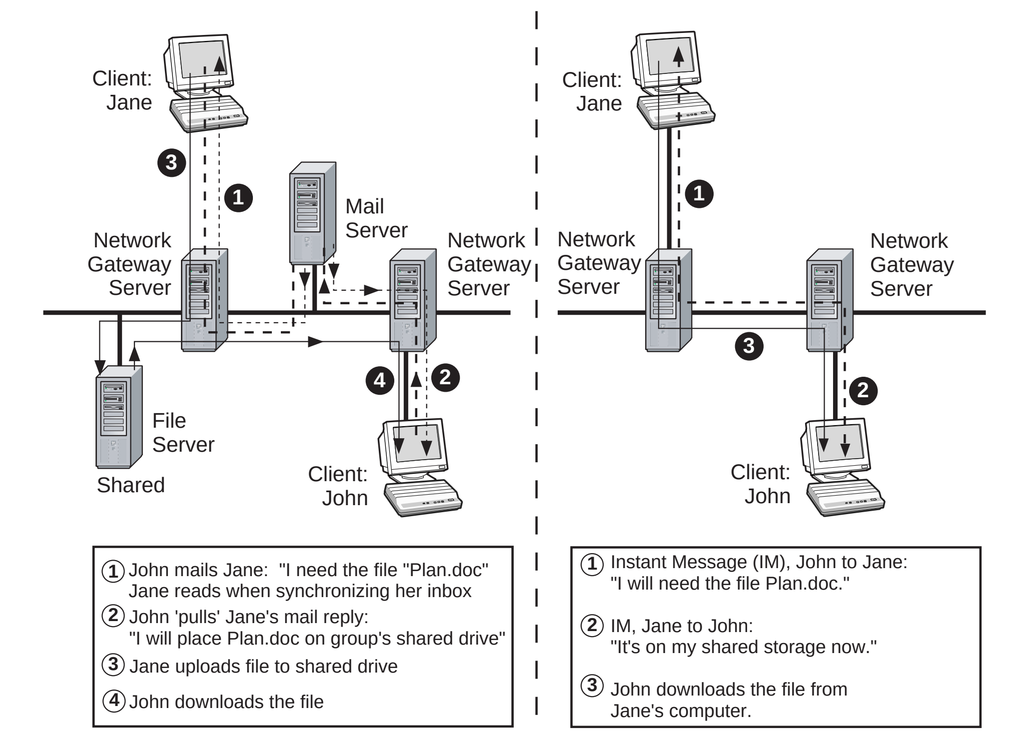


Figure 1.2 Find and Download a File: Client-Server Model and P2P Model

This is a good spot to address, briefly, an issue of definition that is hotly debated among P2P developers:

Some experts make a distinction between *pure P2P* and *hybrid P2P*. The term *pure P2P computing* refers to an environment where all the participating computers are peers. This definition for pure P2P computing looks reasonable, and a number of developers confine themselves to the pure P2P model. However, on closer look, pure P2P is too restrictive. The most obvious example of an application that does not conform to the pure P2P model is Napster. The Napster file exchange occurs between two peers, but the application relies on a server that contains the directory of all files available for sharing from all the peers. This is hybrid P2P computing, where a P2P application uses a server to enable users to interact with each other. P2P computing in this book includes the hybrid model. The role of servers, if any, will be clear from the discussion.

In pure P2P computing no central system controls, coordinates, or facilitates the exchanges among the peers. Note that one or more servers may be needed. Indeed, there is a server layer in each of the participating clients. You may think of each system as being concurrently a client and a server. The use of server here should not be construed as a replacement for a full-blown server. The server layer contains the functions that the applications require for servicing the requests from, and exchanges with, the other peers. An example of the pure P2P approach is Freenet.

Hybrid P2P computing means that the application relies on a central server to perform some of the functions. The degree of central system involvement varies with the application. Napster, for example, requires the user to connect to the control server, where the directory of all available files is held. Once the information is retrieved from the server, the file exchange takes place without central intervention or involvement.

In spite of what the terms imply, no one method is better than the other. Each has its advantages and drawbacks. Each is the right choice for some applications.

Instead of dwelling on a formal definition of P2P computing, it is more productive to look at the purpose of P2P and the actions it enables. The two fundamental actions that P2P enables users to perform are *share* and *collaborate*.

But share what? And collaborate how? Answering these questions helps us understand the purpose of P2P computing in detail. Note that this detail does not, indeed cannot, spell out all the applications that may be possible through P2P. All we can observe now are the building blocks and the mechanisms that can be used. And, admittedly, future advances may show even more building blocks.

What are the resources that can be shared?

- The previous section showed that files and compute-cycles can be shared. Music, video, or any other type of file can be accessed and downloaded from another peer.
- A peer can perform computations on behalf of an application that originated from, and is controlled by, another peer, as we saw with the SETI@home application.
- Storage space can be shared. A peer may provide storage for files of another peer, or for a community of peers.
- Information can be shared by making it available to peers on one's machine or by transferring it to theirs.

- Services can be shared and managed cooperatively by peers. Messaging and security are two examples of such services.

And what methods are available for P2P-based collaboration? This is an area where innovations are evolving, but here are some techniques now used in P2P collaboration:

- Direct exchange between the peers.
- Transactions taking place with the help of an intermediate (as in Napster).
- Granting access to resources (as in SETI@home).
- Any kind of interaction formed through policies that are implied when a peer joins an online community.

One way to derive a definition of purpose that is more inclusive, flexible, and extensible is this: There are *P2P technologies*, and there is *P2P computing*. The P2P technologies enable peer systems to share resources and collaborate on computational and processing tasks. This implies an abundance of supporting technologies and services, such as security (authentication, authorization, etc.), discovery, remote resource management, and more. Any application that takes advantage of P2P technologies is P2P computing.

Note that an application that employs P2P technologies for sharing resources among peers may also utilize a server for some functions, and it may also have some degree of central control. Distributed computing is a technique where many clients (peers) and non-clients collaborate in solving a computational problem. The participating systems share resources, grant access, and contribute to a common goal. However, they do not have to communicate with each other, and the distributed problem is usually centrally controlled. Yet, distributed computing is one of the cornerstones of P2P computing.

In this book P2P computing is the use of P2P technologies. A resulting phenomenon, highlighted throughout the book, is the creation of online communities that collaborate through resource sharing. This is the immediate result and operational purpose of P2P computing.

Several important characteristics are typical of P2P applications. In the previous section you read about some of the better-known examples and early implementations of P2P applications. They used two basic capabilities.

- File-swapping for exchanging content such as music
- Cycle-sharing for collaborative, distributed computations

The attribute of P2P that is highlighted by most of these examples is decentralization. The effect of decentralization is to shift the center of the computation and exchanges to the edge of the net.

But P2P computing is much more. Wonderful environments can be created once we have an infrastructure that provides an easy mechanism for direct and content-rich exchanges between users. Most of the possibilities can be encapsulated by saying that P2P is a tool for creating online communities.

These communities are dynamic. A member may join a group of peers formed with the use of P2P tools for a trial period. Later, that member might invite others, or choose to look elsewhere. In general, the community is *self-organizing*. The group is likely to have no central authority that determines who is a member in the community. Individuals choose to join and to leave. Exceptions to this principle might occur when P2P communities become commonplace inside the corporate world, where membership is more likely to be determined by the associations and functions within the organization. A community may define collectively its acceptable rules of conduct and policies.

The members of any community have a shared interest. This shared interest determines the nature of the exchanges, shared resources, and access between the peers. The principle of *autonomy*, or control of the user over the peer system, should, however, allow each peer to set permissions defining the resources others can access on the peer's computer. For example, an individual may wish to have different permissions for adults and for children.

The online P2P community can also be enduring, in the sense that members can relocate geographically without affecting their contacts and relationships within the online community. As people join such a community it becomes feasible to have an identity that isn't tied in to a physical node and a particular IP address.

The P2P application is particularly *resilient* and *fault-tolerant*, since the P2P application designer needs to assume that peers are not connected all the time. The application has to survive in a very fluid operational environment. In addition, for many P2P applications there is no single point of failure.

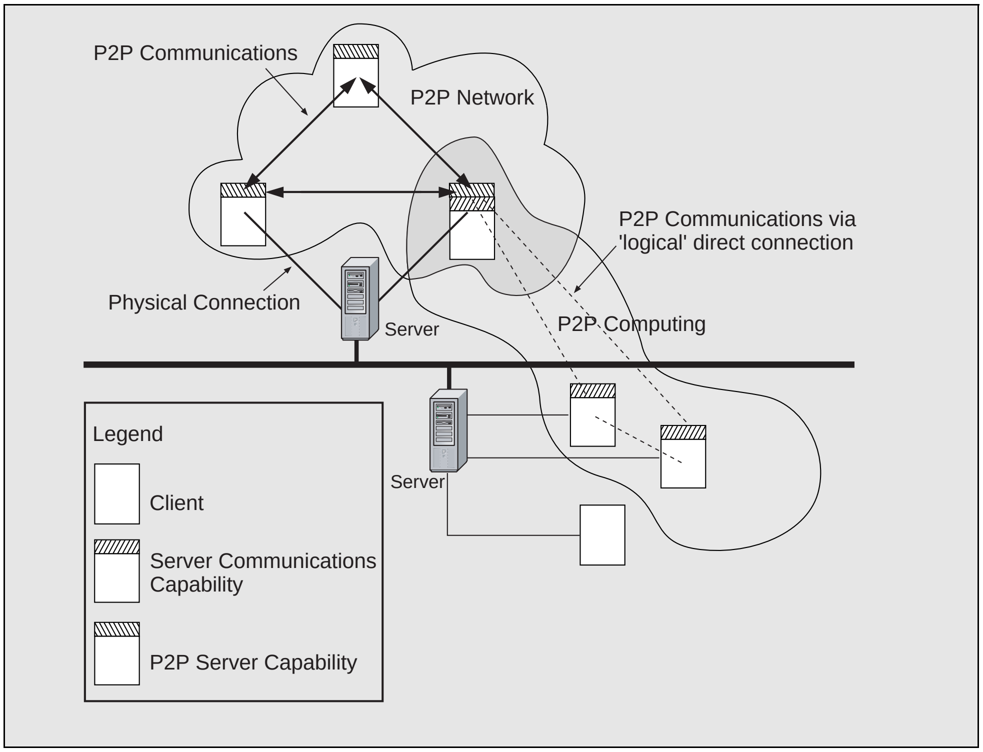
Establishing a P2P community is a collaborative endeavor, which creates an environment conducive to collaborative projects and joint activities. These may be structured projects, such as using the P2P environment for software product development by a virtual team. Or, the P2P community may be a looser assembly of participants who engage in online gaming, for example. The possibilities are endless.

Sharing resources across nodes via direct exchanges creates decentralized online communities, operating at the edges of the net. These communities are self-organizing, dynamic, enduring, resilient, and collaborative.

P2P Computing vs. P2P Network and P2P Communications

Sometimes, to understand a concept better, it helps to know what it is *not*. You may be familiar with the terms *peer-to-peer network* and *peer-to-peer communications*. They are not the same as *peer-to-peer computing.* A P2P network allows every computer in the network to act as a server to every other user on the network. Dedicated servers may or may not be present. P2P communication enables both participants to initiate, manage, and terminate the session. P2P network implies P2P communication between computers in the network. But P2P communication can also occur between two computers in a network that is not P2P. How are P2P network and P2P communications related to P2P computing? Unlike P2P networks, P2P computing does not imply that every computer, or node, can be a server to every other computer. In P2P computing the relationship between users is negotiated in some manner. The participating peers, the computers that can be servers to others, may be a part of a P2P network, or may belong to networks with different characteristics. The kind of services offered in P2P computing varies depending on who the other peer is. Computers in P2P computing have software layers that provide server services. The same is true for all members in a P2P network. The peers in P2P computing are likely to exercise P2P communications; i.e., they have equal authority and responsibility regarding the communications with each other. However, the peers may also agree on communication arrangements that are not symmetrical. For P2P computing, communication is only a means to facilitate sharing of resources.

In short, P2P network and P2P communications refer to the physical network and a mode of communications. P2P computing defines an end-user application-level environment. The relationships among the three concepts are shown below.



What P2P Computing Can Do for You

Most processing and computing results in interpersonal communication, services, and collaboration. Today the road to this person-to-person interaction travels through many intermediate machines—servers, data centers, and firewalls, to name but a few. Add to this the recent popularity and effectiveness demonstrated by consumer P2P applications, and a question naturally presents itself: Wouldn't these technologies be at least as effective within the enterprise? After all, the enterprise provides a more controlled environment than that found in the World Wide Web. The user community is already defined. There are teams who work jointly all the time. And person-to-person or intra-team exchanges are the norm. Peer-to-peer computing belongs in the enterprise.

Of course, P2P can belong elsewhere, too. Witness the success of various forms of file sharing on the net. In fact, most of the development and innovations in P2P technologies are done outside the enterprise by start-up companies and research institutions. The enterprise has the resources and the environment to test, validate and qualify. It is very appealing to combine the creativity emanating from outside the corporate world with the resources and discipline of the enterprise in order to advance P2P computing.

This line of thinking, the synergy between the corporate-enterprise world and the small teams of entrepreneurs, extends to the opportunities of applying P2P services in the fast-evolving e-Commerce arena. Electronic Business-to-Business (B2B) commerce is augmented by online Business-to-Consumer (B2C) commerce. e-Commerce is maturing to provide more comprehensive two-way exchanges between suppliers and consumers, and variants of e-Business allow for Client-to-Client (C2C) transactions. P2P technologies fit in and offer huge opportunities for e-Commerce and the supply chain on which it relies. In summary:

- Peer-to-peer computing belongs in the enterprise.
- The entrepreneurial creativity joining forces with the resource-rich, disciplined enterprise is a recipe for success.
- Peer-to-peer computing offers huge opportunities for the advancement of e-Commerce.

Where do P2P applications for business and consumers thrive? Here are some things that become possible with P2P technologies:

- Community web network: A group with specific common interests—a family, a village, hobbyists, etc. can expand the idea of lists, shared resources, and a web site and create their own intranet.
- New capabilities for e-Business:
  - Connect and enable all the links that make up a complete supply chain.
  - Distribute information, content, or software more effectively in a cascading manner. It is akin to turning a single pipe into an ever-expanding network of connections.
  - Keep details that are needed for e-Commerce but occupy large amounts of storage, or are likely to change in time, on their original node. A central directory or a search capability can refer queries to the source. This is useful for real estate, catalogs, classified advertising, auctions, and many other e-Commerce activities.

- Gaming: A P2P infrastructure provides a natural foundation on which to develop online community games that are not centrally controlled. The developers can focus on the game's features, instead of the interface to the communications protocol.
- Search engines: Overcome the problems of stale data and a single source for answering inquiries by searching directly across the horizon of the community. The usefulness of this techniques requires that the coverage area is adequate for the item being searched for.
- Virus protection: The relationships among the nodes allow the P2P community to collaborate in virus detection and warning, as well as automatically quarantine itself against further attacks (Sandia National Laboratories, 2000).
- Edge services: There are instances when it is desirable to locate the data, prior to its use, closer to the client requesting it. For example, such relocation can assure consistent rate of data transmission for video streaming, whether for entertainment or online web-based training. Multiple clients offering storage space can, arguably, provide a more flexible and reliable service compared to that offered by a server, since a server needs to respond to multiple requests simultaneously.
- Collaborative development: The scope can range from developing software products, to composing a document, to testing, to rendering graphics, and more.
- Large-scale computations: Enterprises and organizations can utilize available computational cycles to solve complex scientific and engineering problems. Such applications may include aircraft simulation, drug design, computational chemistry, climate studies, and many more.

The scope and span of possibilities is exciting and virtually limitless.

The Peer-to-Peer Challenge

For some 20 years now people in academia and research have experimented with aspects of peer-to-peer technologies, mostly in relatively small and controlled environments. There are also some products that enable users to collaborate and share files. And recently we have seen the first applications that caught the imagination of millions of users. The constellations are now aligned. The physical infrastructure of computers and communications is in place, and the concept of P2P in the mainstream of computing has been demonstrated. The time has come to move from the mode of early experimentation to that of a solid computing model.

There are some challenges, however. Today every application or company creates its own basic services, peer plug-ins, and environment.[9] And P2P applications that originate from different sources cannot be integrated and cannot communicate with each other. The application developers do not benefit from existing solutions; they spend time creating the same services instead of developing more interesting and feature-rich applications. The lack of integration results in less than optimal experience for the user. This situation can be resolved when the Industry gets together and works to achieve interoperability between P2P applications.

Interoperability can be accomplished through the use of common protocols. Having agreed on the protocols, there are a number of ways to get to them from the application. First, the application developers can make system calls or employ low-level coding that will exercise the protocol's functionality directly. This is hard and is platform dependent. Second, developers can use a common set of Application Program Interfaces (APIs). This option is much easier for application developers, but is still language and environment dependent. Third, developers can use a common set of services that provide the functionality needed for P2P computing above that which is provided by the peer's operating system. This set of common services acts as a *middleware* layer. One of the advantages of a common middleware is that application developers do not need to keep creating the same basic services over and over again. Given that the common middleware runs on multiple platforms, the part of the application that uses the middleware is also portable to other platforms.

Figure 1.3 shows a P2P middleware layer that fits between the peer's local operating system, and the application interfaces that the P2P application can call directly. Note that applications can use different API schemes when calling middleware services, and still be interoperable.

[9] The only exception to this "one of a kind" is, perhaps, the activity around grid computing for distributed processing. Distributed processing is described in Part II.

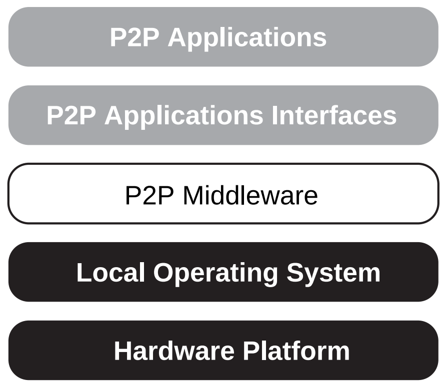


Figure 1.3 P2P Middleware

The challenge, and opportunity, facing the community of P2P advocates is learning to work together in creating an integrated environment where applications can interoperate. The common middleware approach makes it possible to address the P2P infrastructure requirements together. These common functions, some of them quite complex, include security and privacy, access control, fault-tolerance and availability, scalability and extensibility, heterogeneity of operating systems, hardware, policies, and some more you will read about later.

The challenge, then, is to create a framework for solving the common problems together, while allowing plenty of space and flexibility for innovation and creativity.