Editor: Geoffrey Fox, fox@csit.fsu.edu

PEER-TO-PEER NETWORKS

By Geoffrey Fox

WILL PEER-TO-PEER COMPUTING BE THE NEXT "KILLER" INTERNET APPLICATION? LIKE MOST OVER-HYPED CONCEPTS, P2P IS LOOSELY DEFINED AND COVERS A SET OF RATHER DISPARATE IDEAS. PERHAPS THE ONLY COMMON THEME IS a client-oriented view of the world; you might think of P2P as "Power to the People." The servers are subservient to the clients, which do most of the work.

P2P has several important technology challenges and applications, varying from the sublime to the ridiculous. This column presents a quick overview and suggests some emerging research areas and opportunities.

Napster

Napster (www.napster.com) is the most well-known and popular P2P system. Shawn Fanning developed the original application and service in January 1999 while a freshman at Northeastern University. Napster lets any client advertise the MP3 files stored on its disk and download MP3 files from other clients connected to the Napster server network. It is said that Shawn was taking a computer programming course at Northeastern but had to buy a programming book to build Napster. Like most good ideas, Napster was designed to solve a real need—in this case to enable Shawn, a musician, to share his music with his friends on campus. The system has become staggeringly popular. A legal opinion from last summer stated that users were sharing approximately 10,000 music files per second using Napster, more than 100 users tried to connect to the system every second, and there would be 75 million Napster users by the end of 2000 (http://news.cnet.com/News/Pages/Special/Napster/napster_patel.html).

Napster has other typical P2P services—instant messaging, chat rooms, "buddy lists," and information about today's popular music—but the key feature is the ability to share files between any Internet-connected consenting clients. This is roughly the Web version of NFS (Network File System) familiar from traditional computing environments.

Another feature is its handling of MP3 files, which are important as a popular digital encoding for audio files. It is straightforward to "rip" (copy) files off an audio CD and look up key metadata (artist, title, and so on) in a so-called CDDB database on the Web (www.gracenote.com). You can store and access the audio and metadata as a single unit. Although the system uses a server to establish the initial connection, it transfers files efficiently, directly from client to client. This is an improvement over most NFSs where using distributed files is not easy (except possibly for the originator), because usually all you have is a filename. The added value of metadata for files lies at the heart of the Semantic Web, a vision from the W3C Web Consortium related to P2P (www.w3.org/2001/sw).

Although the two concepts of Web-based NFS and metadata-enhanced files are fundamental and broadly applicable, Napster is currently controversial and under siege, because the audio files are typically copyrighted. The company's legal problems are a feature of the particular content, but the technology is long-lasting and in my opinion uncontroversial—it's an essential P2P capability.

Some 200 Napster clones are available to support this area (www.ultimateresourcesite.com/napster/main.htm). Currently the most popular is Imesh (www.imesh.com), which has some two million users and can share any type of file. Some of the best-known file-sharing systems are MojoNation (www.mojonation.net), Freenet (http://freenet.sourceforge.net) , and Gnutella (http://gnutella.wego.com). These are not server-based like Napster but rather support waves of software agents expressing resource availability and interest propagating among an informal, dynamic network of peers.

Other P2P features

So far, we've examined some basic P2P services—file registration, access, and search. We can also categorize P2P systems in other ways, including distributed computing, collaboration, and core technologies. Let's review these three areas.

Distributing computing. Distributed-computing P2P applications are well illustrated in the CiSE article "Distributed Projects Tackle Protein Mystery" by Keri Schreiner in the first issue of 2001. Schreiner discusses the use of millions of Internet clients to analyze data looking for extraterrestrial life (SETI@home, http://setiathome. ssl.berkeley.edu) and the newer project examining protein folding (Folding@home, www.stanford. edu/group/pandegroup/Cosm). This kind of distributed-computing solution divides an application into a huge number of essentially independent computations plus a central server system that doles out separate work chunks to each participating client. In the parallel-computing community, these problems are called pleasingly or embarrassingly parallel. I include this approach in the P2P category because the computing is peer-based, even though it transfers files through peer–server communication (unlike the largely pure client–client model of Gnutella and Napster). SETI@home and Folding@home are elegantly implemented as screen savers that you download.

Other projects of this type include United Devices (www.ud.com/home. htm based on SETI@home), AppliedMeta (based on the University of Virginia Legion project, www. appliedmeta.com), Parabon Computation (www.parabon.com), Condor (from Wisconsin www.cs.wisc.edu/ condor), and Entropia (www.entropia. com). Other applications for this type of system include financial modeling, bioinformatics, Web server performance testing, and the scheduling of different jobs to use idle time on a network of workstations. Ian Foster has given a more detailed review of these activities at www.nature.com/nature/ webmatters/grid/grid.html and relates them to computational grids (www. gridforum.org).

Collaboration. Collaborative systems form a rather different type of P2P network: a community of clients working together and sharing different Internet resources. Instant-messenger (IM) communication and chat rooms, in all their various forms, are the most common examples in this arena. Here, participating clients exchange messages with each other. Unlike the file-sharing case, a user would typically multicast the same message to multiple clients at the same time. The best architecture for this is still under active research. In fact, I work in this area (see the Garnet Collaboration System at http://aspen.csit. fsu.edu/collabtools). Groove Networks (www.groove.net), founded by the creator of Lotus Notes, is the best-known P2P collaboration project; it uses relay servers to implement the P2P multicast. Collaboration systems form a P2P "illusion" using a static or dynamic suite of servers to optimally route messages. When the clients are scattered around the world, the relaying servers would perhaps be in the "middle" of the Web; when a group of clients are clustered, their relay would be "on the edge" and perhaps dynamically created on a peer machine of this cluster. Typically you also need some sort of server to establish the initial session and manage the permanent state. So this type of P2P application gives a rich mix of true peers and servers.

In addition to IM, chat, and email, collaboration systems offer shared resources such as white boards, shared documents, and audio-videoconferencing. The HearMe system is a nice example of the P2P illusion (www.hearme.com). A central server manages digital audio conferences with a general mix of phone and pure Internet audio. As their technology advances, they should move to the Groove and Garnet models with dynamic relay servers positioned throughout the Web. All forms of collaboration use some type of messaging, with the message (typically called an event) carrying a variety of content including the IM text, pixel changes to record a changed shared display (frame buffer), or digital audio packets. XML is the natural way of encoding such messages, and the open source Instant Messenger Jabber (www.jabber.org) provides a clean framework of this kind. Several Napster-like systems have based their service on IM technology; Aimster (www.aimster.com) is one of the best known. OpenCola (www.opencola.com) has a general XML framework to support P2P systems.

Core technologies. Core technologies or services include P2P management, messaging, security, and client grouping, as well as the file (or more generally object) registration, discovery, and access capabilities mentioned earlier for Napster. We must develop these core capabilities and then define community standards. Then they can interoperate, resulting in bigger and better P2P systems. Sun Microsystems has two important technology projects. Jini (www.sun.com/jini) deserves a column of its own; it has a beautiful, simple model for dynamic self-defining objects. Like Napster peers, these objects register with distributed servers so that other peers can discover and access them. JXTA (from *juxta*position, www. openp2p.com/pub/a/p2p/2001/02/15/ joy_keynote.html) is a new project from Sun's chief scientist Bill Joy aiming at core P2P capabilities, including peer grouping and security.

AD

Besides the messaging services needed to implement collaborative P2P systems, there is also a Java message service called JMS (http://java.sun.com/products/jms) that provides the core publish–subscribe mechanism on which most P2P services are built. This needs some upgrading to join the P2P revolution; Sun should add XML, a more dynamic matching paradigm (of collaborating peers), and support for relay servers.

I expect research and commercial experience to identify more base services as we better understand the common needs of P2P systems. Resource management in such a network must be an important challenge. It is our Nirvana—the Web Operating System. Maybe society can live in a hodgepodge of unstructured knowledge swept back and forth by armies of Gnutella agents, but this will not do for the Enterprise P2P (named by the Gartner Group (www.oreillynet.com/pub/d/547) and O'Reilly) needed by Fortune 500 organizations. Here we will need to manage structured information within a dynamic P2P grouping.

The right approach is to generalize Napster and Jini, ensuring that all objects are tied to metadata (possibly in a separate record) that define their discovery, rendering, access, and sharing characteristics. One homely example is family photos; usually these are indeed a melange of folders haphazardly stuffed in shoeboxes. This gets much worse for a community event recorded in shoeboxes across the nation. With the proper metadata and Enterprise P2P support, such photos could be nicely organized and presumably of greater value.

The future

Many interesting ideas are being explored. As an example, breaking shared files into many parts could increase bandwidth (parallel I/O) and security (no one site could access files without cooperation from its peers). This type of technology is controversial, because it makes censorship very hard. MojoNation has a load-balancing and scheduling algorithm in the form of micro payments to reward those who contribute most to the community. Gnutella, a family of related products, is usually described as a P2P search engine, because its interface is more like a search engine than a Web file system.

There is one more important characteristic of P2P networks: client heterogeneity. Hand-held devices, cell phones, special interfaces for those having physical handicaps, as well as basic desktops can participate in a single P2P session. This requires that each peer be able to render copied files and shared objects differently. Careful design of the XML metadata for both clients and display devices makes this quite possible and ripe for research.

There are lots of good research topics and obviously lots of business opportunities. P2P networks, as part of the next wave of the Web, are intellectually challenging to design and socially and intellectually rewarding to use. They can and will unite us all.

You can find general discussions on P2P technology at two good Web sites: www.openp2p.com from the O'Reilly group and www.peer-to-peerwg.org from an industry working group originally initiated by Intel. I also recommend a remarkable book, Peer-to-Peer: Harnessing the Power of Disruptive Technologies by Andrew Oram, Nelson Minar, Clay Shirky, and Tim O'Reilly (O'Reilly & Associates, 2001).