BROADCASTING VIA INTERNET

Broadcasting on the Internet is a relatively new practice. Most of us now take for granted the full, multimedia-rich experience that the World Wide Web provides. But this is actually the result of an explosive growth in interest about the Internet in general and the possibilities the World Wide Web has as a new communications medium in particular. This in turn has driven research and development at a breakneck pace.

The Internet was initially created as a way of sharing information quickly and efficiently between institutions involved in research projects for the Department of Defense. Over the years, this was expanded into a worldwide "network of networks" communicating via a standardized protocol known as Transmission Control Protocol/Internet Protocol (TCP/IP). With the appearance of Hypertext Markup Language (HTML) and browsers such as Mosaic and later Netscape, a graphic front end to the World Wide Web was born. In addition to text, users could see pictures and download files of all types, including audio and video. But these large files took a long time to download over slow modem connections, even at reduced quality.

In an effort to reduce or even eliminate these download times, file size reduction schemes were developed. These algorithms drastically reduce the amount of data contained in a file. Some of these schemes are lossless, meaning the original files can be completely reconstructed from the compressed versions. For multimedia files the data reduction has to be so drastic for practical purposes that the decompression results in an approximation of the original file. These algorithms are generically referred to as codecs (*co*der/*dec*oder). Once the file sizes of audio and video became manageable, it was only a matter of time before broadcasting in real time became a reality.

The earliest attempts to use the Internet as a broadcast medium occurred in 1992, when the Internet Engineering Task Force (IETF) broadcast audio and video from two of their meetings using the MBONE, a subset of the Internet specifically configured to accomplish the task at hand. These broadcasts were successful as a proof of concept, but required a lot of bandwidth and some fairly sophisticated computer programming skills to view. Still, they were the genesis of what was soon to become a worldwide race to turn the Internet into a first-class platform for broadcasting purposes.

The first commercial broadcasting solution was RealNetworks' (formerly Progressive Networks) RealAudio system. Launched in April 1995, it delivered voice grade audio programming in real time over connections as slow as 14.4 kbit/s modems. Instead of downloading a file, the RealAudio Player immediately played the audio information that was being sent to it across the Internet. It was followed that August by Xing Technologies' Streamworks, which used Moving Pictures Expert Group (MPEG) compression techniques. Streamworks offered both audio and video streams, though the video streams required substantially faster Internet connections.

The first public, live broadcast occurred on September 5th, 1995 when RealNetworks broadcast a Seattle Mariners–New York Yankees baseball game. Within the next two years a host of other companies would launch audio and video streaming solutions, and thousands of websites would be offering programming both live and archived. Though there are a number of different streaming media solutions, they all share a similar basic architecture, which is known as the Client–Server architecture.

CLIENT–SERVER ARCHITECTURE

In its simplest form, the client–server architecture consists of a client, which requests a file from a server. The client can be a particular machine or program running on a machine, such as a web browser or a streaming media player. The server, in turn, is a dedicated piece of either hardware or software that processes requests from various clients. Typically during Internet broadcasts a large number of clients will be talking to a number of different servers to distribute the load and to provide redundancy.

Using this architecture, a user would request audio/visual programming by clicking on a hyperlink in their web browser or selecting a preset in their streaming media player. Their browser or player, the client, requests the appropriate file from the media server. The server locates the file and breaks it into data packets that can be sent or "streamed" across the Internet. When these packets arrive moments later, the player then reconstructs the programming being streamed and sends it to the appropriate output device—that is, the speakers and/or the screen. The key point here is that the programming never has to touch the user's hard drive; the whole process happens in real time in the computer's random access memory (RAM).

The client is generally some sort of player interface, either a separate pop-up application or built right into the user's web browser. Clients built into browsers are known as plug-ins or active-X controls. The client generally gives the user some amount of control over the stream, such as volume, play, pause, fast-forward, and rewind (unless of course it is a live stream). The server receives commands from the client and acts accordingly. The functionality of the client is dependent

the media server system being used and the protocol being used to deliver the content.

In large-scale broadcast situations, the media servers will typically reside on separate machines and quite possibly in different physical locations from the web page servers offering access to the programming. This is done for bandwidth considerations and for redundancy. If any one particular machine fails, clients can be redirected to working servers. In this scenario, instead of users requesting programming directly from a media server, they will request content from an intermediate machine which will know something about all available active servers. This machine may also know something about the topology of the Internet and may try to make intelligent decisions about which server to send the client to. This can alleviate bandwidth and traffic. Choosing a geographically closer server or a server with particularly good connectivity can mean less lost information, known as packet loss, and therefore a better signal.

UNICAST VERSUS MULTICAST

These days the most precious commodity on the Internet is bandwidth. Even though the capacity is increasing at an incredible rate, the accelerated growth of Internet usage and bandwidth hungry applications dictates that conservation of bandwidth is paramount. The current model for most broadcasts is such that each audience member gets a unique stream delivered to his or her computer. This method is known as unicasting [see Fig. 1(a)]. For short, static archived files, this poses no problem and is indeed desirable. This ensures that each user will have access to and control over the programming he or she desires. But for live broadcasts this is an extremely inefficient use of bandwidth. Even though the individual streams may be very low bandwidth, when hundreds or even thousands are listening simultaneously, the load can be significant. This can lead to degraded performance for everyone. A much better model is where a single stream is sent out on the Internet and everyone wishing to participate in the broadcast receives a local copy. This is known as multicasting [see Fig. 1(b)].

In a multicast setup, the data packets are sent out over a network and forwarded from router to router, until a specific predetermined time limit is reached, at which point the data are simply discarded. This time limit, known as time to live (TTL), is specified so that the data "live" long enough for everyone to get a copy, but not so long that the network becomes overwhelmed. This is a highly efficient model for local area networks (LANs) where the topology is well known and the routers are easily controllable.

But the Internet was not designed as a broadcast network. Unicast is easy to implement because the routers that control data flow on the Internet were designed to send packets to specific addresses. Current attempts at multicasting have to be "strapped on" using specifically configured servers and routers. This involves the cooperation of a large number of people, specifically system administrators and network engineers. The administrative effort is not trivial. Help is on the way in the form of an IETF multicasting standard that will be built into all future routers. Many routers today are already "multicast-enabled." The proposed standard will also include multiple layers of service, bandwidth reservation protocols,

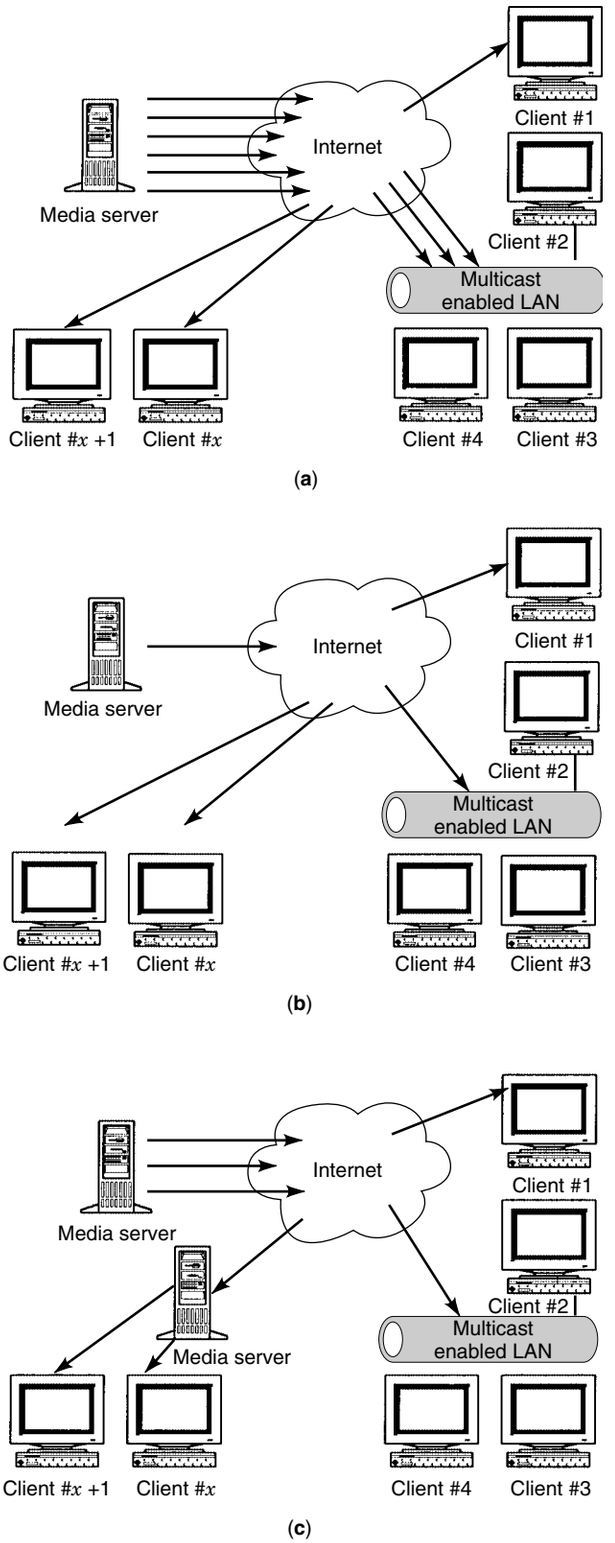

Figure 1. Unicasting, multicasting, and distributed multicasting.

and a host of other essential features necessary for the Internet to become a true broadcast medium.

An interesting hybrid of the two has appeared which is known as distributed multicasting [see Fig. 1(c)]. In this scenario, a master server distributes individual streams to a number of secondary servers. These secondary servers can then either unicast, multicast to local or wide area networks, or split yet again to a tertiary layer of servers. Listeners who are on multicast-enabled networks get multicast streams, and others are steered to unicast servers. This method tries to work within the current limitations of the Internet, taking advantage of multicasting when it can, and unicasting when it must. By keeping the number of streams traversing long distances across the Internet at a minimum, it also tries to minimize bandwidth usage.

DELIVERY PROTOCOLS: TCP/IP, UDP

Regardless of whether the broadcast is going to be unicast or multicast, we have to have a method of ensuring the data gets to where we want it to go. Though sending data across the Internet may seem a simple task, in reality it involves a large amount of cooperation between many different pieces of software and hardware. Each link in the chain must know what it is being handed and what to do with it. It is convenient to think of the process as having different layers. The highest layer would be the client application that the user sees and interacts with. The lowest layer would be the machinery that actually transmits the electrical impulses across the wires that join pieces of equipment. In between these there are still other layers. Communication between each layer uses a known protocol and is, in theory, unconcerned about other layers in the process.

The Internet has developed around the TCP/IP. While there is some disagreement about how many layers it is comprised of, it is convenient to think of it as consisting of four: the application layer, the transport layer, the Internet layer, and the network access layer. Figure 2 illustrates this along with a brief explanation of each. It was designed this way to allow development at any layer to occur without impacting every other layer in the protocol. In this manner, faster hardware or new software can be developed and brought online as long as it adheres to its layer protocol.

The most critical layer for the purposes of this article is the transport layer. The two most common protocols used in this layer are TCP and User Datagram Protocol (UDP). TCP is the mechanism by which web pages are delivered. It is extremely reliable, because it asks for confirmation from the receiving end that every packet has been received. If this acknowledgment is not received, the data are resent. So it is very robust, but it is not necessarily efficient for time-based delivery. Resending the lost or unacknowledged data may take a long time, and by the time it arrives it might be too late for the player to use. By imposing its own flow scheme upon the data, TCP can effectively destroy the temporal relationship between packets. Streaming media is "time-critical," in that you can't fill in the blanks after the fact like you can with an image or a page of text. In addition, TCP has more overhead associated with it, and therefore it is not the most efficient use of bandwidth.

UDP is a much leaner though less reliable protocol. But because it has less overhead and doesn't require the constant acknowledgment messages, it tends to be better suited to time-based delivery. To combat the reliability problem, several methods can be used. Simple error correction methods such as parity bits and checksums can be used. The data can be interleaved, whereby instead of each data packet containing a contiguous "chunk" of media, it will contain $1/x$ of a portion of x "chunks," which are then rearranged properly in the player. In this manner, if any particular packet is dropped, the signal degradation is spread over a longer time interval and not as noticeable. Last but not least, sophisticated error correction methods can be built into the codec that is decoding the incoming data.

Most streaming media applications will use a combination of the two, with the delivery occurring via UDP and the player's control communication occurring via TCP. As with most things on the Internet, an interesting hybrid known as robust

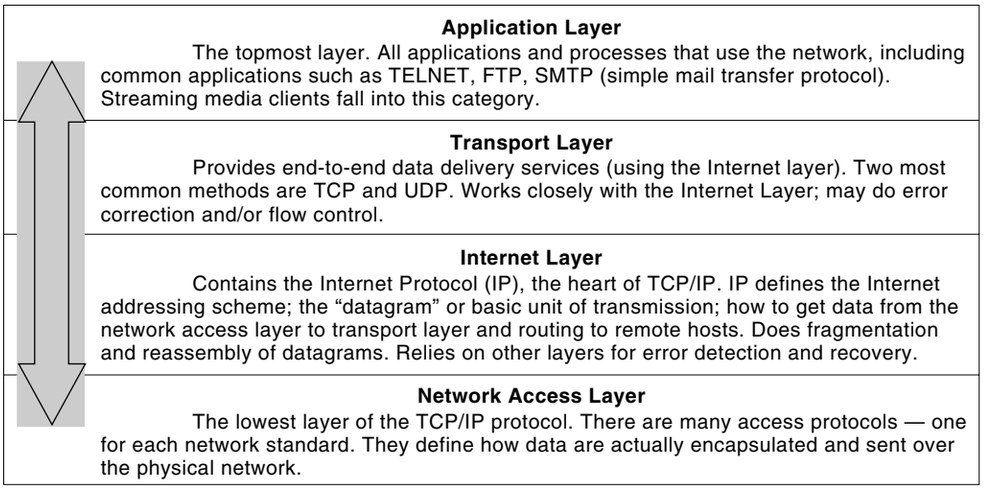

Figure 2. The Internet in terms of its component protocol layers. Each layer is independent and only needs know how to hand data to the next.

Table 1. Some Sample Data Rates and Accompanying File Sizes for Media Files

Media Type	Data Rate	File Size for 1 Minute
Uncompressed video		
640 × 480, 30 frames/s	211 Mbit/s	1.54 Gbytes
320 × 240, 15 frames/s	26.4 Mbit/s	198 Mbytes
176 × 144, 15 frames/s	8.7 Mbit/s	65 Mbytes
Uncompressed audio		
44 kHz, 16 bit stereo	1.35 Mbit/s	10 Mbytes
22 kHz, 16 bit stereo	689 Mbit/s	5 Mbytes
8 kHz, 16 bit mono	125 Mbit/s	938 kbytes

UDP has appeared. It has been noted that with a small increase in the pre-buffer time, a player can request dropped packets from the server via TCP and often receive them via UDP in time for them to be of use.

ENCODING FOR LOW-BIT-RATE TRANSMISSION

The majority of users still access the Internet via a dial-up phone connection, typically at data rates of 28.8 kbits/s. Since raw audio and video generate significantly higher data rates, to make broadcasting in real time over the Internet a reality, a vast reduction in the amount of data has to occur. Table 1 lists some audio and video data rates for comparison purposes. This reduction is accomplished by sophisticated compression schemes. Many of these have been around for awhile and in use, most notably in satellite and telephony applications. Others have been developed specifically for low-bit-rate transmission and storage of audio and video.

The first thing we can do to reduce the data rate is to reduce the screen size and frame rate of the video and to limit the frequency range of the audio. These are effective but not sufficient. The bit rate available to us is so constricted that we must use the more powerful tools of data compression and data reduction. Typically the encoders used for Internet broadcasting will use a combination of methods to achieve the desired results.

Data compression takes advantage of the fact that there is often a large amount of redundancy in digitized media. The most straightforward of these is run length encoding which replaces repeating sequences by a number that indicates the number of repetitions followed by the pattern itself. For instance, the pattern "999999" could be replaced by "69." This is the sort of coding used in fax machines. This method is extremely efficient and simple to implement but typically not powerful enough on its own for use in media applications.

Another data compression method is known as entropy coding, or Huffman coding. By analyzing the data using statistical techniques, a code can be arrived at that assigns (a) a small number of bits to represent the most common patterns in the file and (b) longer codes for patterns that appear less often. These algorithms are complex and can introduce latency to the broadcast. For highly efficient Huffman coding the statistical information has to be known in advance. If this is not known or the statistical profile of one file differs greatly from another, the code can generate more data than the original.

Both run length and Huffman encoding algorithms are lossless, which means that on the playback side the original file can be completely reconstructed using the known code. Using run length and Huffman coding to compress the data, compression ratios from 1.5 : 1 to 3.5 : 1 are possible, but this is insufficient for Internet broadcasting purposes. We must rely on other lossy methods collectively known as data reduction.

In data reduction, perceptual coders are used which take advantage of what we know about how we perceive audio and video. For audio signals, psychoacoustic models are used to determine what we actually hear. For example, a loud crash will hide or "mask" a conversation someone is having at the next table. There are many factors such as amplitude, frequency, time, and location that affect the way we hear and what we actually perceive. For visual information, psychovisual models are used. These take advantage of the limitations of our visual system. Using these models allows us to discard information that may be deemed below the threshold of perception. In addition, video coders take advantage of redundancy between frames and try not to encode areas that do not change from frame to frame. Bringing these sophisticated methods to bear allows us to reduce the amount of data drastically enough to be streamed across the Internet. Using these lossy reduction schemes means that the original file cannot be reconstructed from the data that arrives. The quality of the broadcast will by definition be greatly reduced. However, the number of interested listeners is driving codec research and quality is improving on a near-daily basis.

AUDIENCES, FUTURE TRENDS

The number of people with Internet connections is still growing at a staggering rate. Studies now indicate that the time people are spending on the Internet is often time previously spent in front of their televisions. Though the largest number of simultaneous viewers for live broadcasts currently number only in the thousands, the number of web sites adding audio and video content is growing daily. As these numbers grow and economic models that make the Internet a commercially viable medium are developed, more and more multimedia content both live and archived will become available. Currently every major media company in the United States has a web presence, and every one has some form of multimedia content on their site. The benefits the Internet offers as a delivery medium are simply too great to be ignored. Beyond the fact that physical copies no longer need to be delivered by conventional means, the Internet offers new levels of interactivity for the viewers. Content can be either "pushed" (as in traditional television models where the viewer is passive) or "pulled" (where the viewer actively chooses the content or even interacts with it).

Intellectual property rights are a big area for discussion. When perfect digital reproductions are just a click away, who ensures that the copyright owner gets paid? New technologies known as digital watermarking are being developed specifically to solve this problem. These involve placing encrypted copyright information and licensing details in the stream itself. Streams can then be licensed to specific clients or for specific lengths of time.

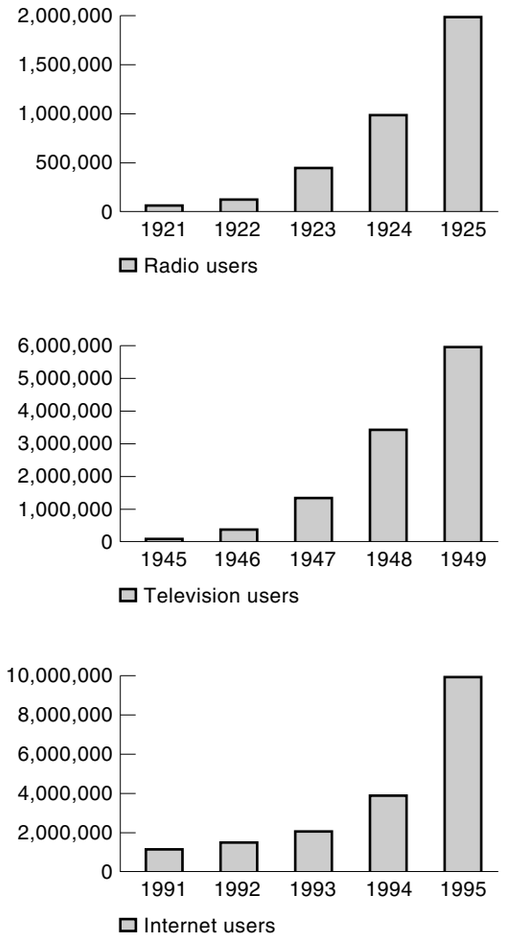

Figure 3. Adoption curves of new mass mediums.

Who could have guessed back in 1992 that the IETF would start a new industry, or at least define a level playing field where anyone can be a broadcaster? What began as an experiment quickly became a working concept that has grown in a few short years into a thriving new business attracting interest at the highest levels. Comparing adoption curves of radio and television to the current Internet adoption curves shows an interesting parallel. From this we can also learn that it takes time before new mass mediums become viable businesses (see Fig. 3).

The infrastructure of the Internet is improving on a daily basis, and new technologies promising more bandwidth to end-users are announced nearly as often. Codec research is bringing in better quality at lower bit rates. Multicasting is soon to become an Internet standard, which will greatly improve and expand the reach of Internet broadcasts. These factors combined point toward an optimistic future where the Internet can become a new mass medium, the first mass medium where the audience can talk back.

BIBLIOGRAPHY

T. Berners-Lee, *The World Wide Web: Past, Present and Future* [Online]. Available www: http://www.w3.org/People/Berners-Lee-Bio.html/1996/ppf.html

S. Heath, *Multimedia & Communications Technology,* Oxford: Focal Press, 1996.

C. Hunt, *TCP/IP Network Administration,* Sebastopol: O'Reilly & Associates, 1994.

K. C. Pohlmann, *Principals of Digital Audio,* 3rd ed., New York: McGraw-Hill, 1995.

B. Segal, *A Short History of Internet Protocols at CERN* [Online], April 1995. Available www: http://wwwcn.cern.ch/pdp/ns/ben/TCPHIST.html

STEVE MACK
RealNetworks

BROWSERS, INTERNET. See INTERNET BROWSERS.

BROWSER WARS. See INTERNET BROWSERS.

BRUSHLESS DC MOTORS. See SYNCHRONOUS MOTOR DRIVES.

BUBBLE MEMORY, MAGNETIC. See MAGNETIC BUBBLE MEMORY.