Media Streaming over the Internet

— an overview of delivery technologies

Franc Kozamernik
EBU Technical Department

This article reviews the basic concepts of media streaming over the Internet, particularly those associated with Internet Protocols (IP) and server technologies. The article concentrates on the delivery aspects of audio and video over the Internet but does not cover audio and video compression schemes nor media capturing/authoring/publishing tools in any detail. These aspects will be covered in future editions of EBU Technical Review.

The concept of streaming media is less than a decade old and yet it has experienced impressive growth. Using streaming technologies, the delivery of audio and video over the Internet now reaches many millions of people using their personal computers – offering *live* sport, music, news, entertainment and *on-demand* content. With broadband networks being deployed in many countries, and video/audio compression technologies advancing rapidly, the quality of audio and video services over the Internet is increasing rapidly. A variety of user terminals can now be deployed, ranging from office desktops to personal digital assistants (PDAs) and mobile phones.

In this article, *streaming* stands for the real-time [1] transport of live or stored media (e.g. video, audio and any associated data) over the Internet, between the client and server computers.

There are two modes for the transmission of media over the Internet:

- In the **download mode**, the user can play the downloaded file only after the whole file has been downloaded from a server to his/her computer. The full file transfer, in the download mode, can often suffer unacceptably long transfer times, which depend on the size of the media file and the bandwidth of the transport channel. For example, if downloaded from **http://www.mp3.com**, an MP3 audio file encoded at 128 kbit/s and of 5 min duration will occupy 4.8 MB of the user's hard disk. Using a 28.8k modem, it will take about 40 minutes to download the whole file [2].
- In the **streaming mode**, however, the content file need not be downloaded in full, but plays out *while* parts of the content are being received and decoded.

1. The term "real time" means that the user receives a continuous stream – near-instantaneously (with a "minimum" delay) – and that the duration of the transmitted and received streams is exactly the same.
2. Throughout this article, the following notation is used: b = bit, B = byte = 8b, k = 10^3, K = 2^{10} = 1024. In general, bits refer to streams and bytes to memory and storage; for example, a stream of 56 kbit/s, a video file of 100 MB or 10 GB.

Fig. 1, adapted from [1], gives an analogy between the delivering of multimedia and the drinking of water.

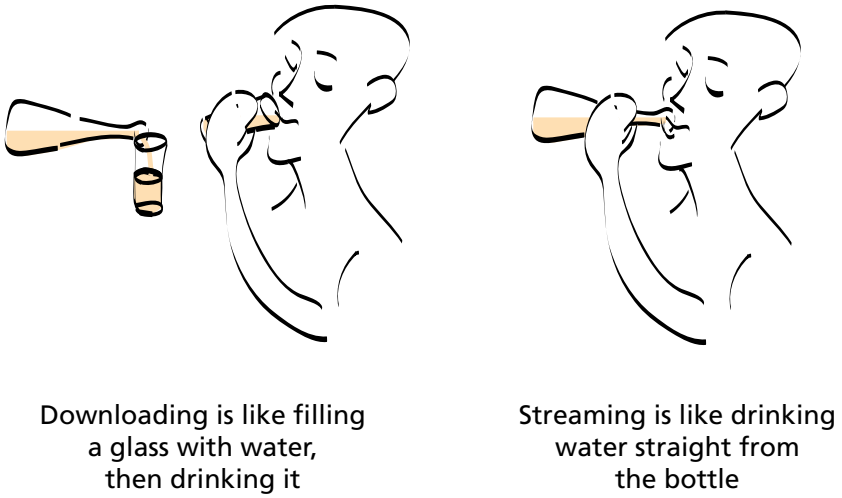

Figure 1
A downloading and streaming analogy with drinking water

There is also a third approach: **progressive download** or **pseudo-streaming**. It is neither streaming nor download-and-wait: the media starts playing a few seconds after the download starts. If the Internet connection speed is faster than the media data rate, the user can watch the video while it is downloading. For the user, it looks like streaming but, in effect, it is downloading. The information that the player needs to start playing the video is stored at the beginning of the file. The media content is interleaved throughout the file in such a way that parts that play together are located together in the file.

The streaming mode is like television. It is broadcast, you watch it and, at the end of the transmission, it is gone. Progressive download is like watching a programme on TV but also recording it on your VCR. After it has been broadcast and taped, you can watch it again and again.

This article does not deal with download, even not with progressive download. Instead, it provides an introduction to *streaming* technology and describes some unique features and applications that have revolutionized the delivery of media over the Internet during the past five years or so.

Abbreviations

ASF	(Microsoft) Advanced Streaming Format	**NNTP**	Network News Transport Protocol
CSRC	Contributing SouRCe	**OSI**	Open Systems Interface
DF	Don't Fragment	**PDA**	Personal Digital Assistant
DSL	Digital Subscriber Line	**PFGS**	Progressive Fine Granularity Scalable
DVB	Digital Video Broadcasting	**PVR**	Personal Video Recorder
FF	Fast Forward	**QoS**	Quality of Service
FGS	Fine Granularity Scalable	**REW**	Rewind
FTP	File Transfer Protocol	**RSVP**	ReSource reserVation Protocol
HTTP	HyperText Transfer Protocol	**RTCP**	Real-Time Control Protocol
IETF	Internet Engineering Task Force	**RTP**	Real-time Transport Protocol
IP	Internet Protocol	**RTSP**	Real-Time Streaming Protocol
ISDN	Integrated Services Digital Network	**SDP**	Session Description Protocol
ISO	International Organization for Standardization	**SMTP**	Simple Mail Transfer Protocol
		SSRC	Synchronization SouRCe
ISP	Internet Service Provider	**STB**	Set-Top Box
LAN	Local Area Network	**TCP**	Transmission Control Protocol
MIME	Multipurpose Internet Mail Extension	**UDP**	User Datagram Protocol
MMS	Microsoft Media Server	**URI**	Uniform Resource Identifier
MPEG	Moving Picture Experts Group	**URL**	Uniform Resource Locator
MTU	Maximum Transmission Unit	**URN**	Uniform Resource Name
NAT	Network Address Translation	**VCR**	Video Cassette Recorder

What is streaming?

Streaming is a relatively new technology; it is less than 10 years old. The development of streaming capitalises on the ubiquity of the Internet, which has led many multimedia parties to explore the possibilities for delivering media over the Internet. Following the success of conventional radio and television broadcasting, research has been carried out into ways of delivering live media (or "broadcasts") over the Internet to a PC. Some EBU experience with live streaming and webcasting is reported in [2] and [3].

Streaming is the only technology that is capable of transmitting video and audio events across the Internet in real time, i.e. while they are happening. Furthermore, multimedia is expanding from the PC onto new user platforms: hand-held wireless devices, interactive television set-top boxes, games consoles, 3G mobile phones, etc.

The developments in streaming were prompted by three factors:

a) advances in compression algorithms for audio and video;

b) developments in streaming servers;

c) improvements in broadband networks (the telcos' "last mile") and in cable modems.

For a content or service provider, some new equipment and facilities are required, such as a streaming server and an encoder/multiplexer. The cost of this equipment and the cost of the network bandwidth are often major obstacles in using streaming.

The following is a summary of the main features of streaming:

- It can deliver *live* content such as a football match, a concert or a political speech – as it happens.
- It provides random access to long movies. The streaming server can act as a remote video player and perform some VCR functions (e.g. skip back and forth, enable watching only a portion of a media production without downloading the whole film).
- It occupies no space on the user's hard disk. The user does not get a copy of the media file – that stays on a streaming server. The user can save a media, but what is actually saved is the URL of the stream, the current point in the media's timeline and the user's settings (such as the sound volume level).
- It only uses the exact network bandwidth it really needs. If the streaming content exceeds the connection speed, some data packets get lost and the content may break up.
- It allows for streaming tracks (e.g. stock and news tickers) to be included in otherwise non-streaming content.
- It can benefit from using broadcast and multicast approaches (where one stream can be sent to many users).

Some caution should be applied to streaming because it does not always go through firewalls and Network Address Translation (NAT).

These features should be contrasted with those of (progressive) download. The latter cannot send live transmissions, cannot skip ahead (the user has to download the whole media), it puts a copy of the media file on the viewer's hard disk (which could be several GB in size), and it does not allow for broadcast and multicast. On the other hand, with download, the media gets through no matter how slow the connection is, and lost packets can be retransmitted. In addition, no special server software is required for download.

Clearly, both streaming and downloading technologies have their merits and should be used for different purposes. They are both here to stay. If we want to deliver live concerts, we will use streaming; if we are to deliver stored movies, we will use progressive download. If people are using fast (broadband) connections, we will use streaming; however, for slow connections, we will use download.

Table 1, adapted from [4], compares streaming and progressive download.

Table 1 — Comparison between streaming and downloading

	Streaming	(Progressive) Download
Server	Streaming server is required	Standard web server is sufficient
Network layer protocol used	UDP/IP	TCP/IP
Application layer protocol used	RTP/RTSP	HTTP
Packet loss	Packet loss acceptable	No packet loss
Time performance	Real time. The delivered media duration is the same as original	Packets may be retransmitted, leading to slower delivery times
Delivery quality	Some packets may be discarded, to meet time and/or bandwidth constraints	High-quality delivery guaranteed, no data is lost or discarded
User connection	Can match the user's bandwidth	File is delivered without regard to the user's bandwidth
Playback	File starts playing immediately	Playback begins when all of (in progressive: enough of) the file has been downloaded
Effort	More burden on service provider (requires server, multiple bit-rate versions and formats)	More burden on the end user (hard drive space, connection speed)
Firewalls	May not play behind some fire-walls	Bypasses most firewalls
Storage	No files are downloaded to the user's PC	Files are downloaded to the user's PC
VCR functionalities	Yes (for streaming of pre-recorded material)	No
Zapping of internet radio channels	Smooth	Not possible

Requirements

Media streaming over the Internet poses stringent requirements on bandwidth, delay and packet loss. The Internet does not provide any guarantees that any media packet will reach its destination, and the packets which do arrive may not have followed the same route – such that they may arrive in a different order to which they were sent. In order to make it suitable for the transport of time-sensitive information, some application-level mechanisms and protocols need to be considered.

Streaming media applications have to be compressed to match the end user's actual connection throughput. As shown in *Table 2*, adapted from [6], the actual connection throughput of a dial-up and DSL connection is generally much lower than the maximum throughput quoted by ISPs and is limited by propagation and noise conditions on the line. Cable modems share the fibre that is connected between the head-end and the user node; therefore, the available bandwidth-per-user will diminish as the number of simultaneous users increases. In ISDN and E-1 networks, the data rate is guaranteed. There is a general rule that media should be encoded at a lower rate than the rated bandwidth of the network that is specified to carry it. Standard encoding tools have headroom already built into their encoding templates.

Table 2 — Actual throughputs and safe bit-rates for different user connections

User connection	Rated bandwidth	Actual throughput	Safe bit-rate
14.4K modem	14.4 kbit/s	9.6 kbit/s	8 kbit/s
28.8K modem	28.8 kbit/s	19.2 kbit/s	16 kbit/s
33.6K modem	33.6 kbit/s	24 kbit/s	20 kbit/s
56K modem	56 kbit/s	40 kbit/s	32 kbit/s
Single ISDN	64 kbit/s	64 kbit/s	64 kbit/s
Dual ISDN	128 kbit/s	128 kbit/s	128 kbit/s
DSL downlink	384 kbit/s	300 kbit/s	240 kbit/s
E-1	2.048 Mbit/s	2.048 kbit/s	2 Mbit/s
Cable modem	6 Mbit/s	2.4 Mbit/s	300 to 1000 kbit/s
Intranet/LAN	10 Mbit/s	3.0 Mbit/s	2 to 3 Mbit/s
100Base-T LAN	100 Mbit/s	10 Mbit/s	6 to 10 Mbit/s

An architecture for media streaming

The architecture for media streaming can be divided into six areas as follows *(see Fig. 2, adapted from [7])*:

Media compression and encoding: For effective streaming, the content data rate must be lower than the user's connection speed, or else the media is not watchable. To put video or audio onto the web, compression ratios of 100 to 1000 are not unusual. Such drastic compression levels may have dramatic impact on the received quality. Generally, the higher the compression level, the lower the subjective quality that is achievable. The

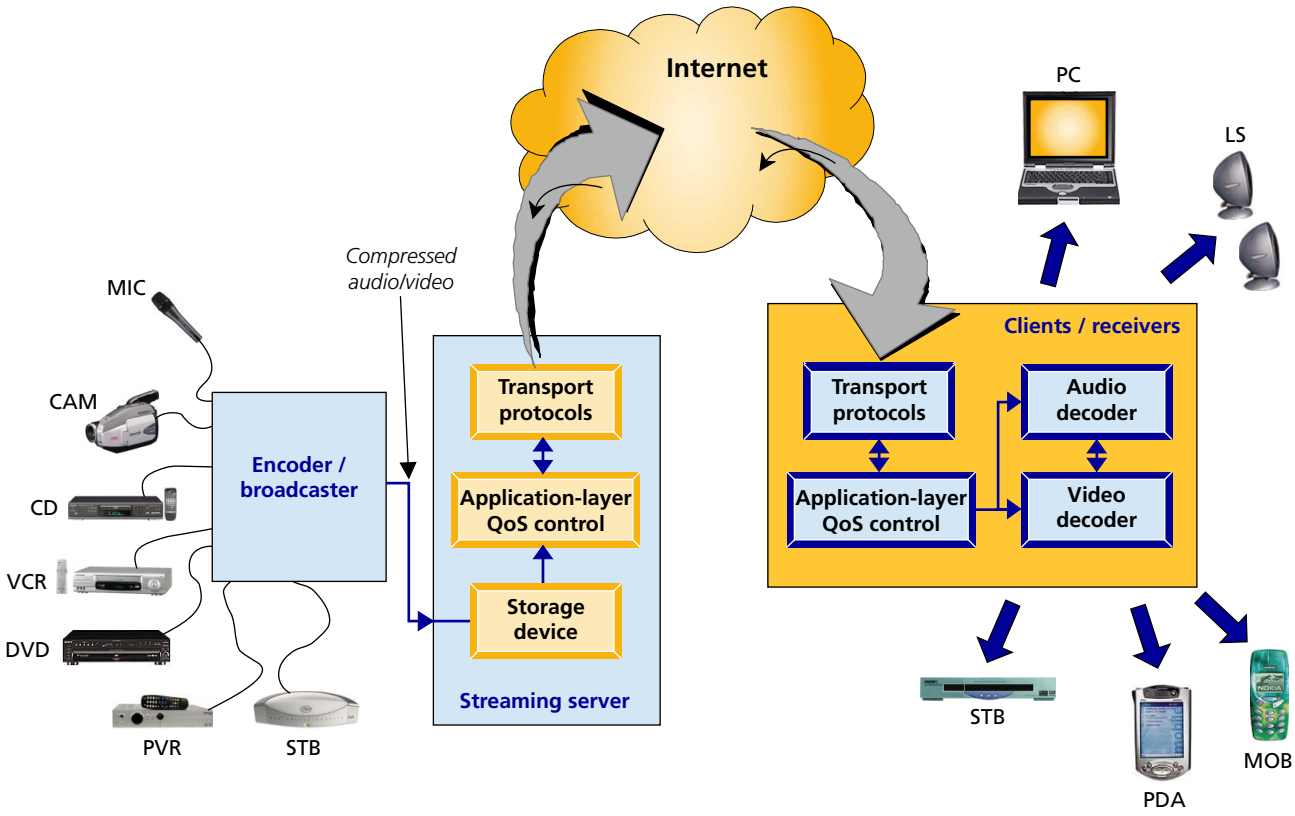

Figure 2
Notional diagram of streaming media

quality depends not only on the bit-rate but also on the content, display resolution, frame rate and the algorithm used. Some compressors allow us to set not only the target data rate but also the colour depth, image resolution, frequency of key frames and other additional characteristics. The most popular compression algorithms on the market are Windows Media, RealNetworks and QuickTime (a vigorous three-way rivalry!); these algorithms can also be streamed efficiently over the Internet, as they are scalable, i.e. they are capable of coping gracefully with the channel-bandwidth fluctuations of the Internet [3].

Movies can be streamed from the server's hard disk. It is possible to stream *live* events, in which case an application program called a *broadcaster* is needed. The *broadcaster* takes input from live sources, such as a video camera, microphone or an ordinary audio CD, compress these sources on the fly for transmission at the desired data rate, and creates a real-time audio-video stream.

Streaming over a LAN requires only a *broadcaster.* In order to stream over the Internet, however, a *broadcaster* should forward its output to a streaming server which is normally located near the Internet backbone. So in all probability the *broadcaster* and the streaming server will be geographically separated. The encoder and *broadcaster* are generally located near the video editing suite, whereas the streaming server could be outsourced to the Internet Service provider (ISP).

Application-layer QoS control: The application-layer Quality-of-Service (QoS) control involves *congestion control* and *error control* which are implemented at the application layer (rather than the network layer). Congestion control is used to prevent packet loss and to reduce delays. Error control is employed to improve the presentation quality in the presence of packet loss. Error control mechanisms may include Forward Error Correction (FEC), retransmission, error-resilient encoding and error concealment.

Media distribution services: In addition to the application-layer support, adequate *network* support is necessary to reduce transport delays and packet losses. The network support involves network filtering, multicast and content replication (caching).

Streaming servers: Streaming servers are arguably the most important single element for providing quality streaming services. Streaming servers process multimedia data under timing constraints and support interactive control functions such as pause/resume, fast forward, fast rewind. The streaming servers are responsible for serving video, audio, slides and other components in a synchronous fashion. A streaming server normally waits for an RTSP request from the viewers *(see the section "Internet protocols" on page 7)*. When it gets a request, the server looks in the appropriate folder for a hinted media [4] of the requested name. If the requested media is in the folder, the server streams it to the viewer using RTP streams.

Media synchronization at the receiver side: The receiver side should be able to present various media streams in the same way as they were originally generated by the streaming server. For example, movements of the speaker's lips should match the played-out audio.

Protocols for streaming media: Streaming protocols are standardized in order to allow for communication between streaming servers and client computers. These protocols specify the following functionalities:

- *network addressing*, which is provided by a network-layer protocol such as Internet Protocol (IP);
- *transport*, which is defined by a transport protocol such as User Datagram Protocol (UDP);
- *session control*, which is provided by the Real-Time Streaming Protocol (RTSP).

These protocols are covered in more detail in the next section.

3. Much of the ongoing work on subjective quality evaluations is carried out by the EBU Project Groups B/AIM (Audio in Multimedia) and B/VIM (Video in Multimedia). For audio results, see EBU document BPN 049 [5].
4. Before streaming, a *hint* track is created for every streamable media track. The streaming server uses the hint track to turn the media into real-time streams *(see the section "Hint tracks" on page 12)*.

Internet protocols

The key to understanding the Internet is its concept of layers. Each layer performs a clearly defined function. Using agreed interfaces, each layer receives data flow from the layer below and delivers the processed data to the layer above.

The Internet is a combination of networks that can carry a variety of different applications. As shown in *Table 3*, each of these applications is characterised by a specific application-layer protocol and uses either TCP or UDP as the *transport* protocol.

Table 3 — Application-layer and transport-layer protocols for different internet applications

Application	Application-layer protocol	Transport-layer protocol
E-mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	UDP
Streaming media	RTSP or proprietary	UDP
Voice-over IP	Proprietary	UDP
Network management	SNMP	UDP
Routing protocol	RIP	UDP
Domain name translation	DNS	UDP

Internet protocols for streaming

Several protocols have been developed and standardized specifically to allow *real-time streaming* of multimedia content over the Internet. Some of these protocols are given in *Table 4*.

Table 4 — Real-time streaming protocols

Acronym	RFC [a]	Title	Notes
RSVP	**2205**	Resource ReserVation Protocol	
	2208	RSVP applicability statement	Guide to deployment of RSVP
	2209	Message processing rules	
RTCP	**1889**	Real-Time Control Protocol	Part of RTP
RTSP	**2326**	Real-Time Streaming Protocol	
RTP	**1889**	Real-time Transport Protocol	Payload type: see RFC 1890 Other fields: RFC 2250
SDP	**2327**	Session Description Protocol	Used primarily in video conferencing
UDP	**768**	User Datagram Protocol	

a. RFC documents are issued by the Internet Engineering Task Force (IETF) as a Request For Comments (RFC) – **http://www.ietf.org/rfc.html**. These documents eventually become *de facto* standards.

Internet Protocol (IP)

IP is the principal network-level (i.e. Layer 3 of the ISO OSI model) communications protocol. IP is simpler than OSI but is essentially an unreliable delivery system. There are three main kinds of drawbacks with IP:

- variable network latency;
- the packets arrive at their destination in a different order from transmission;
- packets may be damaged or lost;

These drawbacks cannot be fixed by IP itself but can be corrected by the higher-layer protocols, i.e. the transport layer and application layer. One of the well-known transport-level protocols is Transmission Control Protocol (TCP) which provides a highly reliable transport for general-purpose data but is not suitable for streaming applications.

Each IP packet is made up of a header (40B) and a payload. The header consists of three parts:

- IP header – 20B
- UDP header – 8B
- RTP header – 12B

There is no requirement for every IP packet in a stream to have the same length.

An example of a basic IP packet format is shown in *Fig. 3* , adapted from [8]:

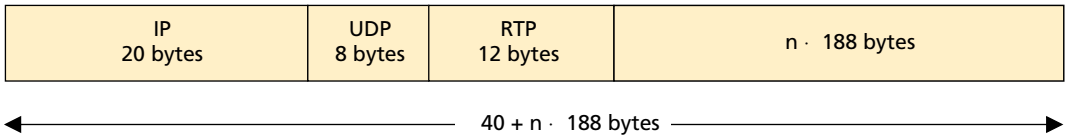

Figure 3
An IP packet example for MPEG-2 encapsulated transport

It should be pointed out that IP packets can be embedded in any physical network support, either cable or wireless, either two-way (communication) or one-way (broadcasting). The network layer that uses IP could be considered a convergence point towards which all media networks may effectively move in the future.

Transport Control Protocol (TCP) and User Datagram Protocol (UDP)

TCP and UDP are two transport-layer (Layer 4) protocols. Although on the same level, they fundamentally differ in the way they are used. *Table 5* shows the main differences between the two protocols:

Table 5 — Comparison of TCP and UDP

TCP	UDP
Connection oriented – a connection is set up prior to data transfer	Connectionless - no connection needs to be established
Reliable	Unreliable
Two-way communication, interactivity between server and client, allows VCR-like functions	One-way only, no interactivity, just start and stop
Error correction FEC	Error detection only – checksum of payload data
Controls data flow to manage the download rate	No flow control
Repeat requests	No repeat requests
No predetermined delivery rate, TCP will increase data rate until packet loss indicates congestion	Delivery rate should match the encoded stream rate. Encoding at several rates may be necessary to adjust to different delivery channels and propagation conditions
Receive buffer overflow: if data arrives too fast, the receiver sends messages to the server to slow down	No local caching – packets are processed by the media player as they arrive

TCP makes a sequence of transmitted bytes and informs the destination of the next byte that the destination is expected to receive. If a byte is not acknowledged within a specified time period, it is retransmitted by the source. This feature allows devices to detect and identify lost packets and request a retransmission. The repeated transmission adds to the latency but this is normally not a major problem in general data transmission.

With audio and video, however, the listener/viewer requires a continuous stream in *real time*. Retransmission of packets adds delays and uses up bandwidth in the data channel. If network transmission errors are high, the receive buffer in the media player will be emptied and the stream will be interrupted. Therefore, the strategy for receiving the streams is to ignore lost packets. The User Datagram Protocol (UDP) does precisely that. The loss of packets may cause impairments to the subjective quality of the received stream or even the loss of several video frames. Nevertheless, media players are often designed to conceal these errors.

HyperText Transfer Protocol (HTTP)

HTTP [9] and FTP are application-level file transfer protocols. HTTP is known as a protocol that effectively carries HTML pages and allows the hyperlinks to transfer the user to another document or web site. HTTP packets are protected with checksums. The server and the client computers have a two-way connection, meaning that there is feedback from the client (receiver) computer. Thus, lost or damaged packets can always be retransmitted, so that the received file can be fully restored. HTTP can also be used for media download, especially if the files are small and the number of concurrent users is limited. If the connection speed is lower than the media data rate, the media still gets through but it may not play smoothly. The transfer time of file download depends on the size of the file and the speed of the connection.

The first version of HTTP, referred to as HTTP/0.9, was a simple protocol for raw data transfer across the Internet. HTTP/1.0, as defined by RFC 1945, improved the protocol by allowing messages to be in the format of MIME-like messages, containing metadata about the data transferred and modifiers on the request/response semantics. However, version 1.0 does not sufficiently take into consideration the effects of hierarchical proxies, caching, the need for persistent connections, and virtual hosts. To this end, the new specification referred to as “HTTP/1.1” was established.

HTTP/1.1 includes more stringent requirements than HTTP/1.0, in order to ensure reliable implementation of its features. It allows more functionality than simple retrieval, including search, front-end update and annotation. HTTP allows an open-ended set of methods that indicate the purpose of a request. It builds on the discipline of reference provided by the Uniform Resource Identifier (URI), as a location (URL) or a name (URN), for indicating the resource to which a method is to be applied. Messages are passed in a format similar to that used by Internet mail as defined by MIME. HTTP is also used as a generic protocol for communication between user agents and proxies/gateways to other Internet systems, including those supported by the SMTP, NNTP, FTP and other protocols. In this way, HTTP allows basic hypermedia access to resources available from diverse applications.

Real-time Transport Protocol (RTP)

RTP is described in RFC 1889 and RFC 2250. This transport protocol was developed specifically for streaming data across IP networks. RTP is the most important streaming standard [5]. All media streams, regardless of their format and content, are encapsulated in RTP packets. RTP provides several data fields that are not present in TCP, in particular a Timestamp and a Sequence Number. RTP runs on UDP and uses its multiplexing and checksum functionalities. It allows control of the media server so that the video stream is served at the correct speed. The media player is then able to reassemble the received RTP packets into the correct order and play them out at an appropriate speed.

5. In addition to the internationally agreed RTP protocol, there are some proprietary data formats used for data transport between the media server and the browser client, e.g. RealNetwork’s *Real Data Transport* (RDT).

RTP transmits packets in real time. Lost or damaged packets are not retransmitted. There are strategies for the client software on how best to cope with the missing bits (e.g. error concealment, packet replication, etc). If the connection speed is lower than the data rate of the media, the transmission breaks up and the media plays poorly (or does not play at all). If the connection is fast, the extra bandwidth remains untouched and the server load only depends on the number and bandwidth of the streams.

Service providers should take care not to exceed the Maximum Transmission Unit (MTU) of the network by setting the DF (Don't Fragment) bit in the IP header to "1". The MTU size depends on the type of network. For instance, for Ethernet-based networks, the MTU is 1500 B. RFC 791 requires that all hosts accept datagrams of up to 576 B. Exceeding the network's MTU leads to a process called "IP fragmentation" which splits RTP packets into smaller packets. Fragmentation should be avoided because if one fragment is lost, the receiving terminal discards all the remaining fragments. Fragmentation also puts an additional burden on the routers that are performing fragmentation and on the receiving terminals that are re-assembling the fragments. Also, some terminals may not support fragmentation.

RTP Header Format

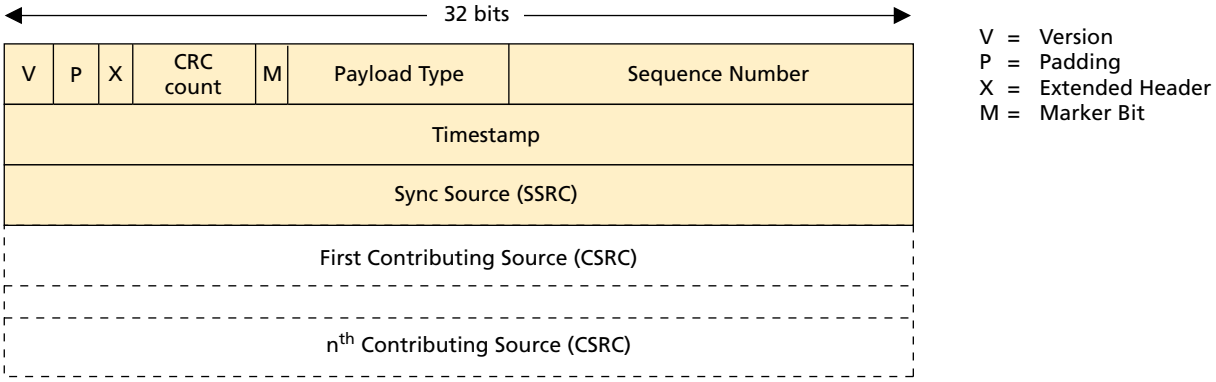

Figure 4
RTP header format

Sequence Number: 16 bit
The value of the Sequence Number increments by 1 for each successive packet. It is used by the player to detect packet loss, to re-order out-of-order packets and to delete duplicates. The initial number for a stream session is taken at random.

Timestamp: 32 bit
This is a sampling instance derived from the transmitter's 90 kHz clock reference and is synchronized to the system's program clock reference. It allows for synchronization and jitter calculations. It is monotonic and linear in time.

Source Identifiers: 32 bit
This is a unique identifier for the synchronization of the RTP stream. One or more contributing sources (CSRCs) exist if the RTP stream carries multiple media streams such as, for example, a mix of several video and audio sources.

Real-Time Control Protocol (RTCP)

RTCP is used in conjunction with RTP and uses TCP for bi-directional client-server connection. It provides feedback to the service provider on the network reception quality from each participant in an RTP session. The messages include reports on the number of packets lost and the jitter statistics (early or late arrivals). This information can be used by higher-level applications to control the session and improve the transmission; for example, the bit-rate of a stream could be changed to combat network congestion.

Real-Time Streaming Protocol (RTSP)

RTSP is an application-level protocol for the control of real-time streaming data. It uses RTP as the underlying data delivery protocol and offers a VCR-like control to the user: Play, Stop, Pause, FF and REW, as well as random access to any part of the media clip. RTSP also helps the server to adjust the media bandwidth to the network congestion in order to suit the available capacity. Another important function of RTSP is its ability to choose the optimum delivery channel to the client. For instance, if UDP cannot be used (some corporate firewalls will not pass UDP), the streaming server has to offer a choice of delivery protocols – multicast UDP or TCP to suit different clients.

RTSP is similar to HTTP/1.1 in terms of syntax and operation but differs in several important aspects. With RTSP, both the client and the server can issue requests during interaction, as opposed to HTTP where the client always issues the requests (for documents).

Microsoft has developed a proprietary protocol for handling the client interaction, using VCR-like controls. This protocol is called MMS (Microsoft Media Server). It uses TCP as the delivery layer. The media data itself can be delivered separately over UDP or TCP in the Advanced Streaming Format (ASF).

Streaming media servers

Streaming media is encoded in streaming files. Apple's *QuickTime* calls them "movies". In order to stream these files in real time, they must be encoded in one of the appropriate streaming formats. All these formats carry timing control information that can be used by the server to manage the flow rate. These files also include indexes that help the user to navigate. There are several streaming file formats on the market. The Microsoft advanced streaming formats have *.wma* and *.wmv* extensions, RealNetworks uses *.ra* and *.rm*, and QuickTime "hinted" movies use the *.mov* extension.

Server stream distribution

Unicast transmission means sending one stream to each receiver. Unicast does not represent a particularly efficient use of bandwidth but it allows the users, by using RTSP functionalities, to watch different parts of the media or watch different movies at the same time. Viewers normally open a unicast media by opening an RTSP URL.

Broadcast transmission (in the networking sense) means sending one copy of the stream over the whole network. A single stream is sent to all hosts in the network. Most small LANs support *broadcast* but the Internet does not allow it. Broadcast does not allow viewers to control the streams (i.e. no VCR functionalities). There is no feedback from the user to the server.

Multicast transmission means sending exactly one copy of the stream, not over the whole network (as in *broadcast*) but only down the branches of the network where one or more viewers are tuned in. In this way, the available network bandwidth can be used more efficiently. Multicast requires fairly sophisticated router software that allows the server to replicate streams as required by the clients. The user of a multicast has no control over the media presented. Like in *broadcast*, the choice is simply to watch or not to watch. The user's host (i.e. PC, PDA, etc.) communicates with the nearest router (rather than with the streaming server) to get a copy of the stream. Users can find out information on multicast programmes by downloading the Session Description Protocol (SDP) file. An SDP file may be embedded in a service provider's web page.

Web servers versus streaming servers

Conventional web servers can only *download* a media file over the Internet but are not capable of *streaming* it. They do not support interactive VCR-like control of the streams either. They ensure accurate delivery – but timely delivery of the packets is not guaranteed. Web servers use HTTP to deliver HTML pages and their associated image files.

Streaming media has the opposite requirements: the delivery must be in real time, but reasonable levels of transmission errors can be tolerated, provided the required QoS can be achieved.

It is technically possible to use a *web server* to deliver streaming files. However, there will be no control over the stream delivery speed. If network congestion is high, the delivery speed will be low; if the network capacity is high, the packets may arrive in bursts. Such a delivery (HTTP) requires a large cache in the client receiver to buffer the incoming packets before the media is played out.

A *streaming server* is a kind of media transmitting device that provides the delivery of live webcasts, webcasts of pre-recorded material, and on-demand (interactive) streaming media. Compared with a web server, a streaming server has the following additional functions:

- real-time flow control;
- intelligent stream switching;
- interactive media navigation.

One major feature of streaming servers is *skip protection*. It uses excess bandwidth to buffer ahead the data, faster than real time, on the client machine. When packets are lost, communication between the client and the server results in re-transmission of only the lost packets, thus reducing its impact on network traffic.

The QuickTime streaming server [10] can serve up to 4,000 simultaneous streams from a single server. To meet increased traffic, multiple servers may be added. The QuickTime streaming server does not require *per-stream charge*. It thus represents an attractive option, as it implies significant savings for customers with high volume media-distribution requirements.

Hint tracks

Apple has introduced the notion of *hint tracks* which the streaming server uses to turn the media file into real-time streams [10]. This concept has been adopted by MPEG-4. For every streamable media track in the movie (e.g. audio or video track), a separate hint track is created to control the stream delivery. The hint track gives the streaming server pointers to the RTP information needed to serve the relevant media chunks.

The movie file container may contain several *tracks*. A track represents, for example, a video or audio object associated with control information that references these media objects. This means that the same video object could be referenced in different movies. The same media file could be repurposed for different pipelines (different bit-rates) or a variety of different displays (resolutions), or indeed multiple languages. For example, the same MP3 file could be created in three different quality levels for streaming at 20, 40 and 100 kbit/s.

Adapting to network congestion

As network congestion varies with the number of users on the network, the available bit-rate for streaming delivery varies significantly. For the optimum viewing and listening experience, the streaming server has to adjust itself dynamically to deliver the highest bit-rate possible at any given moment. Such dynamic adjustments are possible by using the RTCP reports from the media player to measure the network congestion and switch the stream rates to multiple bit-rate media files.

There are two technologies that allow several different encoding bit-rates to be combined into a single media file. The first one is RealNetworks' *SureStream* and the second is a similar technology from Windows Media called *Intelligent Streaming*. At the beginning of playback, the server and the player negotiate to select the most suitable bit-rate. During playback, the player and server communicate repeatedly in order to switch between the different bit-rate versions, thereby delivering the highest quality that the viewer connection can support at any given time. It is important to note that the switching of audio and video is handled independently. The user can even indicate his/her preference whether the change should involve audio or video or both [6].

6. Over very slow connections, the server-side measures are not sufficient to maintain media playback continuity. To this end, the media player can also drop some frames. If the worst comes to the worst, RealPlayer can omit the video track entirely and simply play the audio track.

The drawback of SureStream is two fold: the composite file takes the space of the sum of all the component elements, and a non-Real server will serve the file in its entirety. Only a Real-compatible server is capable of extracting the correct components from the file bundle and then streaming only the best rate for the current network conditions. SureStream and Widows Media Encoder come with predefined multiple bit-rate profiles, but the bit-rates can be customized to suit the user's special requirements. If one wants to multicast a file that has been coded at multiple bit-rates, only the highest bit-rate will be transmitted.

These technologies help the users to continue viewing while the network conditions change, with only graceful degradation in quality. However, they work efficiently only with unicasting.

Discovering the streaming content

In order to serve a streaming file, the streaming server should receive a request from the media player. In order for this request to occur, the following prior steps are necessary:

Step 1 The user locates a streaming media hyperlink on a website and clicks on it. The hyperlink contains the URL of the content. It also contains the instruction to start the media player installed on the user's machine.

Step 2 The URL does not point directly to the content but to a small file on the web server. This file is called the stream redirector – or ASX (in Windows) and Metafile (Ram file) in RealNetworks.

Step 3 The stream redirector metafile is sent to the user's browser via MIME and is passed to the user's player. The metafile contains the full address and filename of the streaming content.

Step 4 The media player then transmits the request to the specified streaming server for the media file via RTSP.

Step 5 The media file is delivered via RTP.

The metafile can list several files to be played in sequence.

Video codecs for scalable streaming

The previous section showed that some scalability can be performed at the streaming-server level. This section shows that by using scalable video codecs, a streaming server will require even less processing power and simpler rate control, compared with non-scalable codecs. Scalable codecs may potentially eliminate the need for coding the content in multiple bit-rates in different formats, and bundling them in the same streaming file. At the same time, scalable codecs are highly adaptable to unpredictable bandwidth variations caused by heterogeneous access technologies or due to dynamic changes in the network conditions. These codecs are resilient to packet losses, which are quite common over the Internet and wireless networks.

It follows from the above that scalable video coding is highly desirable for future streaming video applications. Unfortunately, conventional scalable codecs, using hybrid motion-compensated prediction and DCT coding, lack efficiency. This is due to the recursive structure of the prediction loop.

A new development is underway at several laboratories [11] on the so-called *fine granularity scalable* (FGS) MPEG-4 video codec. This approach may be useful, especially if pre-recorded streams are stored on a server for later streaming. A variation of FGS is *progressive fine granularity scalability* (PFGS) [12]. PFGS shares the good features of FGS, such as fine granularity bit-rate scalability and error resilience. Nevertheless, PFGS is generally more complex; it needs multiple (more than 2) layers as references for motion prediction.

Summary and conclusions

Today, in terms of using streaming media to capture and disseminate our cultural artefacts, we are at the same stage of evolution as the painstakingly hand-copied ecclesiastical manuscripts were, when compared to modern mass-produced paperback books and e-books [1].

Streaming media is a relatively novel technology that enables the Internet user to experience multimedia presentations on the fly. Since its introduction in the early 1990s, the concept of streaming media has experienced a dramatic growth and has became a viable commercial proposition in a relatively short time. This technology is likely to continue to change the way we experience not only the Internet but also media delivery in general. Indeed, many believe that "*streaming media will be the biggest thing since television*" [1].

It is certain that streaming may fundamentally change the way we produce, distribute and consume media. Streaming media is likely to change the economics of production and distribution profoundly. It can embrace radio and television and yet, at the same time, could offer much more flexibility to the user.

The main purpose of this article has been to introduce some basic components on the streaming-server (delivery) side, including the new streaming protocols that facilitate streaming delivery over the Internet. Many related topics have not been covered or even mentioned, including the means of improving Internet delivery by using caching, distributed servers and multicasting. The area of multimedia codecs and players has not been touched upon either.

These, and many other technical and non-technical areas, will be covered in one or more follow-up articles.

References

[1] Michael Topic: **Streaming Media Demystified**
McGraw Hill, 2002
http://books.mcgraw-hill.com/cgi-bin/pbg/007138877X?

[2] EBU document BPN 035, November 2001: **EBU Webcasting Handbook**
http://www.ebu.ch/tech_bpn_035.html *(available to EBU Members only)*

[3] Franc Kozamernik: **A snapshot of the EBU's webcasting activities**
EBU Technical Review No. 291, July 2002.

[4] Jon R. Luini, Allen E. Whitman: **Streaming Audio: The Fezguys' Guide**
New Riders Publishing, May 2002.
http://www.newriders.com/books/

[5] EBU document BPN 049, September 2002: **The EBU Subjective Listening Tests on Low Bitrate Audio Codecs**
http://www.ebu.ch/tech_bpn_049.html *(available to EBU Members only)*

[6] David Austerberry: **The Technology of Video and Audio Streaming**
Focal Press, 2002
http://www.davidausterberry.com/

[7] Dapeng Wu, Yiwei Thomas Hou, Wenwu Zhu, Ya-Qin Zhang, Jon M. Peha: **Streaming Video over the Internet: Approaches and Directions**
IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 1, February 2001.
http://www.ieee.org/organizations/pubs/transactions/tcsvt.htm *(not very helpful!)*

[8] DVB-IPI, doc 016rev: **Transport of DVB services over IP-based networks, Part 1: MPEG-2 Transport streams** *(a working document of the DVB Forum)*

[9] Hypertext Transfer Protocol (HTTP), RFC 2068
http://www.ietf.org/rfc/rfc2068.txt

Franc Kozamernik graduated in 1972 from the Faculty of Electrotechnical Engineering, University of Ljubljana, Slovenia. Since 1985 he has been with the European Broadcasting Union (EBU). As a Senior Engineer, he has been involved in a variety of engineering activities, ranging from digital audio broadcasting and audio source coding to the RF aspects of the various audio and video broadcasting system developments. In particular, he contributed to the development and standardization of the DAB and DVB systems.

Currently Mr Kozamernik is the co-ordinator of several EBU research and development Project Groups including B/AIM (Audio in Multimedia) and B/BMW (Broadcasting of Multimedia on the Web). He is also involved in several IST collaborative projects, such as SAMBITS (Advanced Services Market Survey / Deployment Strategies and Requirement / Specification of Integrated Broadcast and Internet Multimedia Services), Hypermedia and S3M.

Franc Kozamernik was instrumental in establishing the EuroDAB Forum in 1994 to promote and roll out DAB, and acted as the Project Director of the WorldDAB Forum until the end of 1999. He represents the EBU in Module A of the WorldDAB Forum. He is also a member of the World Wide Web Consortium (W3C) Advisory Committee.

[10] Apple: **QuickTime for the Web for Windows and Macintosh**
Second Edition, Academic Press, 2002.
http://www.academic-press.com/

[11] Jens-Rainer Ohm: **Fine granularity scalable coding in MPEG-4**
Proceedings of International Conference on Media Futures, Florence, May 2001, p.177-180
(On CD-ROM, via http://leonardo.telecomitalialab.com/conferences/mediafutures.htm)

[12] ISO/IEC JTC1/SC29/WG11: Text of ISO/IEC 14496-2/FDMA4: **Streaming Video Profiles**
Document No. N3904, Pisa, January 2001.
http://www.iso.ch/iso/en/ISOOnline.openerpage