https://doi.org/10.26552/com.C.2012.2.33-41

Lourdes Beloqui Yuste – Hugh Melvin *

A PROTOCOL REVIEW FOR IPTV AND WEBTV MULTIMEDIA DELIVERY SYSTEMS

This paper reviews the key protocols used for multimedia delivery both over privately owned managed IP Networks such as IPTV and public non-managed IP Networks such as WebTV. Within these two worlds, the choice between protocols is based on the characteristics of the multimedia service required, the media server and the client's receiver.

Users have different expectations of these two delivery platforms and also of different output devices such as TV, PC, tablet or mobile phone. In an IPTV environment, which is a paid-service, users demand the Quality of Service (QoS) they pay for; different quotas provide different services to users. On the other hand, with WebTV, which is a free service, clients have lower quality expectations.

There are multiple multimedia delivery systems used across both platforms, using different protocols to deliver multimedia from server to one or multiple clients. These include Real-Time Protocol (RTP), Real-Time Control Protocol (RTCP), Real-Time Streaming Protocol (RTSP), Real-Time Messaging Protocol (RTMP), Hypertext Transfer Protocol (HTTP), HTTP Live Streaming (HLS) and Microsoft Smooth Streaming Protocol (MS- SSTR).

RTP, RTCP and RTSP are the main protocols used in IPTV while RTMP, HTTP, HLS and MS-SSTR are protocols used mainly for WebTV. This paper will explore the main differences and similarities between them and the reasons behind the choice of one or the other. This review paper also briefly outlines a testbed being developed by the authors to synchronise media streams using a subset of these protocols.

Keyword: *Index Terms–IPTV, WebTV, Media Streaming, RTP/RTCP, RTSP, HTTP, MS-SSTR, RTMP*

1. Introduction

Multimedia traffic over IP Networks includes services such as Internet Video, Voice over IP (VoIP), IPTV, WebTV and Video Calling [1]. Such data transmission requires specialised protocols to accomplish the user quality requirements which for some services includes hard real-time delivery.

The growth of IP Networks for media delivery is increasing day by day. Cisco White Paper [1] and [2] provided interesting data that charts the increase IP Network use and the increase of Internet traffic dedicated to Video. Fig. 1 depicts the 2010 broadband traffic by application subcategory [1] and Fig. 2 shows the global consumer video 2010-2015 by category per month [2]. According to [2], projected data growth over IP Networks between 2010 and 2015 will be 55%, comprising a 32% increase of fixed Internet, and 24% of Managed IP and 92% mobile Internet. Internet video traffic in 2010 was 40% of total traffic although Cisco also forecast that this will reach 62% by 2015. Even geographical areas traditionally behind in Internet Video such as The Middle East and Africa will grow by 105%. Interesting data published in [1] details how Internet Video tripled in 2010 and will grow 17-times by 2015.

In 2010 Internet Video watched on a TV set traffic was 7% of Internet video traffic, this is projected to increase to 16% by 2015 [2]. The same source predicts that Video-on-Demand (VoD) will triple by 2015. In 2011 VoD High Definition (HD) video will surpass Standard Definition (SD) and it is projected that 77% of VoD traffic will be HD by 2015.

The growth of IP Network traffic and the increase in traffic dedicated to multimedia applications (both hard real-time and soft real-time) has led to major developments in protocols and technologies used for multimedia delivery. Multimedia transmission has differing characteristics to other data such as email, etc., such as high network load, real-time delivery and other quality performance criteria.

The delivery of multimedia over IP Networks is typically analysed by two parameters, QoS and Quality of Experience (QoE). The former is based in objective network analysis using metrics, such as delay, jitter and packet loss, whereas the latter is based on the user's subjective opinion about the final media play-out.

Real-time delivery is the main characteristic which distinguishes media delivery from others. Many Internet applications require lossless delivery of information via Internet, such as email where users want to receive the exact content sent by the sender. Multimedia delivery on the other hand is more tolerant of loss but

* **Lourdes Beloqui Yuste, Hugh Melvin**
College of Engineering & Informatics, National University of Ireland, Galway, Ireland, E-mail: lbeloqui@gmail.com

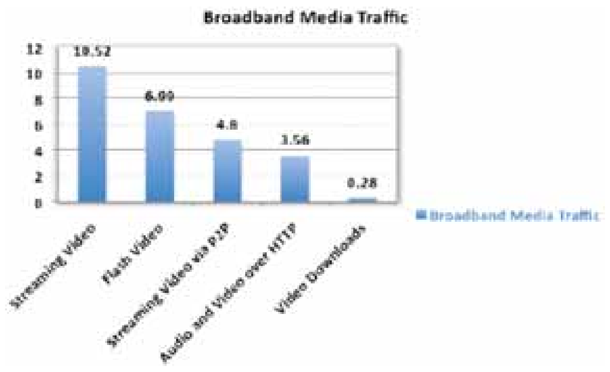

Fig. 1 2010 Broadband Traffic by Application Subcategory. A total of 26.15% dedicated to Online video [1]

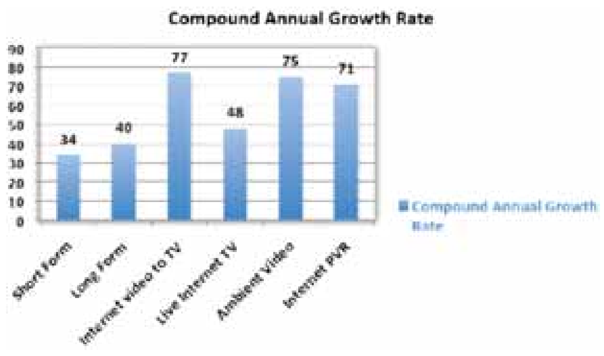

Fig. 2 The Global Consumer Internet Video 2010-2015 by Category per month [2]

WEBTV and IPTV differences Tab. 1

Symbol	webTV	IPTV
Hardware	PC	TV and STB
Software	Browser-based	Media player
	HTTP media selection	Electronic Program Guide (EPG) channel selection
Network	Multiple protocols	RTP/UDP
	Public	Private
	Unmanaged	Managed
	Worldwide	Local
	Mainly unicast	Mainly multicast
	Best effort quality	QoS guaranteed
Media	Unprotected	Protected by encryption
	Multiple coding	SDTV/HDTV
	Access to all Internet media	Limited to IPTV content
	Free media	Paid provider media
User	High level involvement	Low level involvement
	Unsafe, unknown users	Safe, authenticated users
	Free access	Access only to known users
	Free service	Monthly payment

requires that data is received within a time range; otherwise it is not valid. For example, for VoIP and video conferencing, M2E (Mouth to Ear) delays typically should not exceed a round trip time delay of 300msec. Often, relative arrival delays, rather absolute delays, are more important e.g. when we are watching a video over the Internet if the sound related to the image we are watching arrives much later it does not make sense to play it and thus it should be ignored. On the other hand it is better to receive an image of lower quality than one out of synch with audio.

There are two main factors that determine which protocols are used, the media platform and the Transport Protocol used underneath the Application Protocol.

Within the TCP/IP protocols stack, all layers have a big roll to play. Application Layer (AL) protocols are highly dependent on the Network and Transport Layer (NL/AL) protocol used. In this paper we examined protocols mostly at Application and Transport layers and how they are related. Media traffic can use IP unicast or multicast at the Network Layer and User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) at the Transport Layer.

There are two main IP Network delivery platforms, IPTV and WebTV. Due to their different characteristics each one uses different application layer protocols. While IPTV uses RTP combined with RTSP, WebTV is mainly HTTP oriented. HTTP is not a media delivery protocol but the main protocols used in WebTV are based on HTTP media delivery.

The remainder of this paper is structured as follows. Section II depicts the media delivery platform, Section III describes the different media delivery systems such as downloading, progressive downloading, streaming and adaptive downloading. Section IV presents the IP, Transport and Application protocols related to media delivery. Section V briefly describes the testbed developed, and finally Section VI concludes the paper.

2. Media Platform

The media distribution over IP Networks can be performed via two different systems, WebTV and IPTV. Although both use IP they have different characteristics that affect the media delivery protocol used. The main differences are established by Maisonneave in [3] and they are depicted in Table I. An important fact from the user's perspective is their expectations based on cost. Users paying a monthly quota for e.g. for IPTV expect a high QoS and QoE and they expect to receive the services they pay for. Users using free WebTV will accept poorer quality media delivery.

The other aspect is the protection of the media delivery which guarantees media providers that the media will not be illegally

TCP and UDP differences Tab 2

TCP [4]	UDP [5]
Connection Oriented	Connectionless
Guaranteed delivery of data	No guaranteed delivery of data
Flow control provided	No flow control provided
Error mechanisms provided	Basic error mechanism via check sum
Lost packets retransmission	No lost packets retransmission
Eliminated packet loss	Reduces packet delay

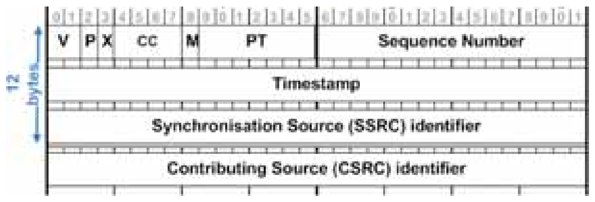
V
P
X
CC
M
PT
Sequence Number
Timestamp
Synchronisation Source (SSRC) identifier
Contributing Source (CSRC) identifier
12 bytes

Fig. 3 RTP Header [13]

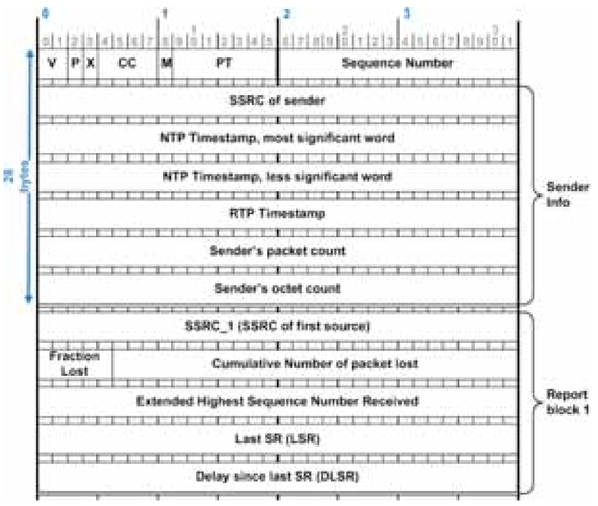
V P X CC M PT Sequence Number
SSRC of sender
NTP Timestamp, most significant word
NTP Timestamp, less significant word
RTP Timestamp
Sender's packet count
Sender's octet count
SSRC_1 (SSRC of first source)
Fraction Lost
Cumulative Number of packet lost
Extended Highest Sequence Number Received
Last SR (LSR)
Delay since last SR (DLSR)
Sender info
Report block 1
28 bytes

Fig. 4.a. Sender Report (SR) RTCP Packet Header [13]

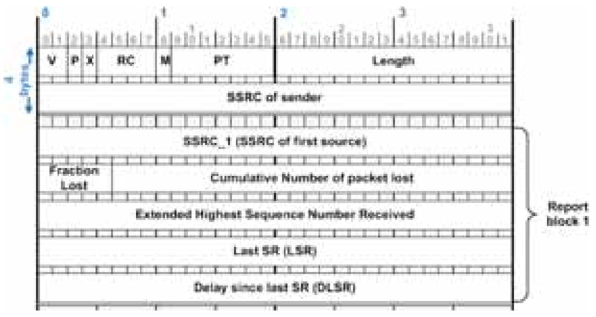
V P X RC M PT Length
SSRC of sender
SSRC_1 (SSRC of first source)
Fraction Lost
Cumulative Number of packet lost
Extended Highest Sequence Number Received
Last SR (LSR)
Delay since last SR (DLSR)
Report block 1
4 bytes

Fig. 4.b. Receiver Report (RR) RTCP Packet Header [13]

copied or distributed. Without network protection IPTV companies would never be able to buy the rights to distribute the media content which is the core of their business.

Both systems use IP delivery but the IP protocol used in each case is different. IPTV uses IP Multicast for video delivery, only employing IP unicast for VoD whereas WebTV mainly employs IP unicast protocols.

3. Media Delivery

There are a few different techniques to deliver and play a media file delivered from the source to the receiver. Different techniques range from downloading and progressive downloading to adaptive streaming and streaming. Downloading and streaming are the most extreme methods whereas progressive downloading and adaptive streaming are hybrid techniques which keep the advantages of downloading and streaming techniques while striving to avoid their drawbacks.

The transport protocol used in the media delivery is a key factor also. The most common transport protocols are TCP [4] and UDP [5]. TCP provides retransmission of lost-packets to guarantee integrity. However it does not assure on-time delivery, whereas UDP provides a low-overhead protocol to provide, quicker, albeit best-effort delivery with no loss-packet retransmission. Table II provides the main differences between both transport protocols revealing the main functionality of UDP to reduce packet delay and traffic overheat to benefit Real-Time delivery.

UDP is typically used by RTP/RTCP, described in Section IV.A, for streaming media delivery. This contrasts with HTTP based protocols, such as HLS and MS-SSTR explained later, which use TCP for downloading, progressive downloading and adaptive streaming methods. Although RTMP uses TCP at Transport Layer it does not employ HTTP for the media delivery.

IPTV systems use streaming to deliver TV multicast services, although they can provide unicast VoD services via HTTP, whereas WebTV usually employs unicast progressive and adaptive HTTP streaming.

A. *Downloading*

The downloading method requires the download of the media file to the receiver prior to its display. The advantage is that once the file is downloaded at client-side the play-out is not interrupted. On the other hand the biggest drawback is the waiting time for the media file to be completely downloaded. The most popular protocol to download files is HTTP based in TCP. Its main drawback is the latency until the video play-out begins. As such it is mainly used for small videos with low quality where latency is minimised [5].

B. *Progressive Downloading*

HTTP Progressive download was developed to reduce the latency of initial play-out, while still preserving all the advantages of HTTP over TCP. HTTP Progressive Downloading is an optimised HTTP technique to stream media over TCP. Media play-out begins before the media file is completely downloaded to minimise the user's waiting time before being able to start playing the video. Multiple web media providers such as YouTube use progressive downloading [6].

RTSP State and methods Tab. 3

State	Message received	Next state
Init: Server initial state and client waiting setup reply		
Init	Setup	Ready
	Teardown	Init
Ready: Server setup, play, pause sent		
Ready	Play	Private
	Setup	Managed
	Teardown	Local
	Record	Recording
Playing: Client has received a play reply from server		
Playing	Play	Playing
	Pause	Ready
	Teardown	Init
	Setup	Playing
Recording: Server recording and client received record reply		
Recording	Record	Recording
	Pause	Ready
	Teardown	Init
	Setup	Recording

C. *Streaming*

The principal objective of streaming is the media delivery at real-time when the main purpose is the media delivery within a time threshold. The underlying protocol used is UDP because it provides a low overhead and minimises packet delay delivery. On the other hand this is also its main drawback as UDP is often blocked by firewalls and penalised by Internet providers.

Streaming benefits include minimising the buffer size at client, low latency, no need to store media file at client-side, efficient use of the bandwidth, and providing the means to analyse the media selection user's behaviour, thus making it the most appropriate delivery method for live video [6] [7].

D. *Adaptive Streaming*

HTTP adaptive streaming uses progressive downloading to deliver the same multimedia content adapted to each individual client. Once the media session has been initialized, on client request or based on network conditions, media can be adapted, for example, to different bit-rate or quality [7].

4. IP/Transport/Application Protocols

The Application protocol used for real-time media delivery depends on a range of factors, including the media platform, IPTV or WebTV, IP Protocols, IP unicast or multicast, and the Transport protocol, TCP or UDP. In IPTV the main protocol is RTP over UDP using IP multicast whereas WebTV utilizes HTTP based protocols using IP unicast.

IP multicast is widely used for IPTV service providers to deliver channels to large numbers of clients with IP unicast used for their VoD services. WebTV, although it can utilize IP multicast, is mainly delivered over IP unicast.

HTTP is used by many companies to develop their own HTTP based Application protocol. Apple is currently finalising the Internet-Draft HLS and Microsoft has created their own MS-SSTR. All of these protocols use HTTP progressive download techniques although they are clearly differentiated. Adobe Flash has developed RTMP which is also based on TCP but not based on HTTP.

Typically the relationship between protocols is dictated by their functions. RTP provides functionality similar to HTTP, both actually delivering the media data, and RTSP relates to RTMP, HLS and MS-SSTR because they organise the complete media session and manage information about the network conditions, the media, the server and the client.

A. RTP/RTCP/RTSP

The benefits of media transport using RTP have been widely studied; [8], [9] and [10] focused on MPEG-2 media streams whereas in [11] and [12] the benefits of the use of RTP on MPEG-4 media streams are explained.

RTSP and RTP/RTCP are mainly used in IPTV systems. RTSP in conjunction with RTP, and its companion RTCP, provide a protocol suite for real-time data delivery such as video and audio. RTP

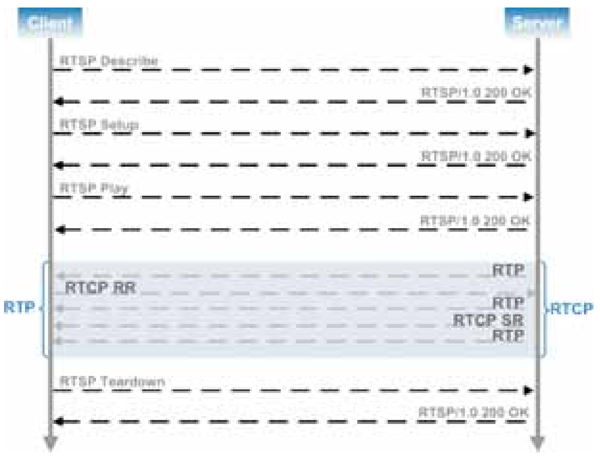

Fig. 5 RTSP Communication [14]

transports the media data, audio and video, while RTCP monitors the media delivery and RTSP controls the delivery of the real-time data.

RTP header provides three important fields for the real- time media delivery which are *timestamp, sequence number and payload type* (PT). The first has a different meaning depending of the payload type, the second helps to reorder the packets at client-side due to the fact that RTP, via UDP, does not provide ordered transport delivery. Finally, the *payload type* is also important as it indicates the media type conveyed within the RTP packet. In Fig. 3 the RTP packet header is depicted.

Once the RTP transport of real-time media streams is initiated, RTCP packets are sent between media session senders and receivers to monitor the media delivery. The media sender sends RTCP Sender Report (RTCP SR) packets and receivers send RTCP Receiver Report (RTCP RR) packets. Those RTCP report packets are different. In Fig. 4a the RTCP SR Packet Header shows important fields such as *NTP timestamp, RTP timestamp* whereas Fig. 4b shows the RTCP RR important fields such as *accumulative number of packet lost, last SR and delay since last SR* [13].

Fields such as *inter-arrival jitter, delay since last SR* and *cumulative number of packet lost* provide information to monitor the media transmission within a media session. The *inter-arrival jitter* provides the inter-arrival time variance of the RTP data packets, *delay since last SR* indicates the time in seconds between the reception of the last two RTCP sender packets, and finally *cumulative number of packet lost* conveys the quantity of packets lost from a source since the beginning of the media session [13].

RTSP maintains the state of the media session. Furthermore with the states and methods, RSTP provides the tools to provide remote control functions over IP Networks. The state is needed to relate a media stream to an RTSP request. States for both client and server are Init, Ready, Playing, and Recording. Not all methods provoke a change of state, *options, announce, describe, get-parameter,* and *set-parameter,* only provide information about the media session whereas *setup, play, pause, teardown, redirect,* and *record* invoke a change of state. Table III depicts the change of states and a description of the client and server states [14].

Fig. 5 depicts the RTSP communication process previous to the RTP media data transport, which includes RTCP packets interchange between client and server, and the media session finalisation. It is important to note that while RTP is transmitted using UDP, RTSP packets are usually sent using TCP to mitigate packet loss.

The process sends an RSTP *Describe* method to the server to require information about the media session. The server responds sending the relevant information via Session Description Protocol (SDP) [15], which is outside the scope of this paper. The client receives media information and sends to the server an RTSP *Setup* command which is responded by server with an RTSP *OK* message. After the initiation of the media session is finalised the client sends an RTSP *Play* and the server finally responds and begins the media transmission via RTP packets. During RTP transmission RTCP SS and RS packets are sent by the sender and receiver respectively. When the client desires to finish the media session sends an RTCP *Teardown* command to the server which accepts with an RTSP *OK* response.

B. RTMP

Adobe Flash Platform technology utilizes RTMP to stream audio, video and data over TCP. This method is designed to stream flash encapsulated media between server and client. It is used in the Internet to transmit both live Internet Radio and WebTV.

As an introduction to RTMP a few concepts such as message stream, chunk and chunk stream need to be clearly specified.

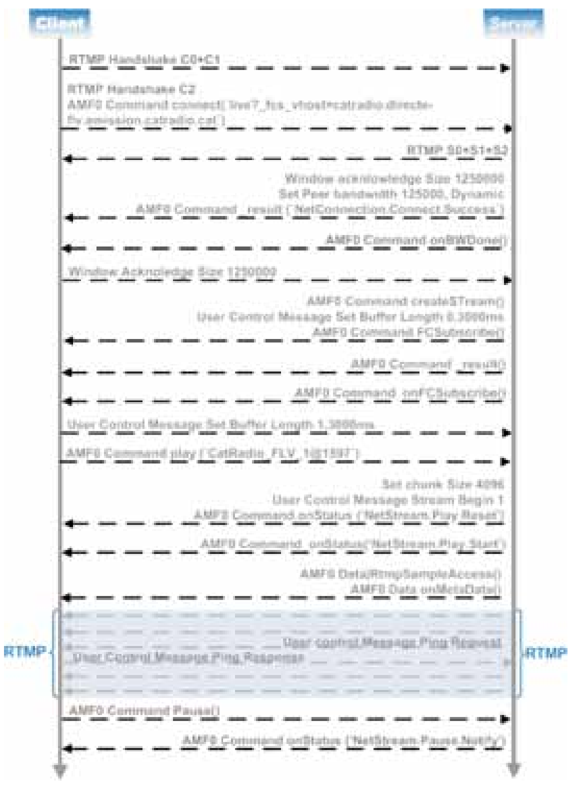
Client
Server
RTMP Handshake C0+C1
RTMP Handshake C2
AMF0 Command connect('live7_fcs_vhost=catradio.directe-flv.emission.catradio.cat')
RTMP S0+S1+S2
Window acknowledge Size 1250000
Set Peer bandwidth 125000, Dynamic
AMF0 Command _result ('NetConnection.Connect.Success')
AMF0 Command onBWDone()
Window Acknoledge Size 1250000
AMF0 Command createSTream()
User Control Message Set Buffer Length 0,3000ms
AMF0 Command FCSubscribe()
AMF0 Command _result()
AMF0 Command onFCSubscribe()
User Control Message Set Buffer Length 1,3000ms
AMF0 Command play ('CatRadio_FLV_1@1597')
Set chunk Size 4096
User Control Message Stream Begin 1
AMF0 Command onStatus ('NetStream.Play.Reset')
AMF0 Command onStatus('NetStream.Play.Start')
AMF0 Data|RtmpSampleAccess()
AMF0 Data onMetaData()
RTMP
User control Message Ping Request
User Control Message Ping Response
RTMP
AMF0 Command Pause()
AMF0 Command onStatus ('NetStream.Pause.Notify')

Fig. 6 RTMP Communication connection to http//www.catradio.cat/endirecte

The message stream is defined in [16] as 'A logical channel of communication that allows the flows of messages'. Messages are fragmented and interleaved to send over the network. A message stream can be an audio, video or data stream.

A chunk is a message fragment that facilitates timestamp ordered delivery, from server to client, of the complete message. Also in [16], a chunk stream is defined as 'A logical channel of communication that allows flow of chunks in a particular direction'.

HTTP versus RTP streaming Tab. 4

HTTP streaming	RTP streaming
Firewall friendly	Firewalls block UDP traffic
HTTP traffic allowed by Network providers	Network providers block or penalize UDP traffic
HTTP-based web server	Specialised media streamer server
Pseudo real-time delivery	Real-time streaming
Pull-based	Push-based
Lack rate control	Rate control
Content stored at client	Content buffered at client
No efficient bandwidth use	Efficient bandwidth use
Longer latency	Minimal latency
Guaranteed content delivery	Unreliable content delivery
Minimises packet loss	Minimises packet delay

Chunks are parts of the audio and video streams, called audio/video messages streams, delivered to client and reconstructed into messages. Small messages can be sent within a unique chunk, while a message's partition into chunks is required when messages are bigger than the maximum allowed chunk size within the message.

RTMP specification consists of the RTMP Chunk Stream Protocol and the RTMP Message Protocol. The former details the message format to convey message chunks and the initial handshake to establish the multimedia connection/session, whereas the latter describes the RTMP message formats, RTMP control and command messages.

Two main concepts are part of the RTMP, NetConnection and NetStream. The former creates a client-server connection whereas the latter represents the communication channel used to deliver audio, video and data streams. Commands are applied to both in a media session. NetConnection commands are *connect, call, close,* and *createStream* while NetStream commands are *play, play2, deleteStream, closeStream, receiveAudio, receiveVideo, publish, seek,* and *pause*.

RTMP first of all establishes a NetConnection via a handshake, initiating interchange of packages between client (packets C0, C1 and C2) and server (S0, S1, S2). Secondly the server provides the client with information about the media session such as Window acknowledge size, bandwidth, set chunk size and buffer length. The final step is to play the NetStream indicated in the transaction.

The message header has four fields *Message Type* (1 byte), *Length* (3 bytes), *Timestamp* (4 bytes) and *Message Stream ID* (3 bytes). This message header is used to transmit control and command messages as well as to initialise a stream message. The chunk header is more complex, being composed of four fields *Basic Header, Chunk Message Header, Extended Timestamp*. The complexity comes from three types of Basic Header plus four types of *Chunk Message Header*.

In Fig. 6 the communication workflow to access a live Internet Radio media is depicted. First the RTMP handshake between client and server is performed via transmission of C0, C1, C2, S0, S1, and S2. Then information about the netConnection and netStream is interchanged and finally the media data transmission is performed via RTMP chunks. Finally the connection client sends a pause command to the server which acknowledges the message.

C. HTTP

HTTP was initially conceived and designed to deliver static documents and consequently rate control was never provided and the underlying use of TCP, with retransmission of lost packets, causes variation in media delivery [17].

HTTP streaming has numerous advantages over RTP streaming mainly caused by the protocol used at the transport layer, TCP. The main differences from [6], [7] and [17] are depicted in Table IV which explains the relative popularity of media delivery via HTTP streaming, particularly for WebTV.

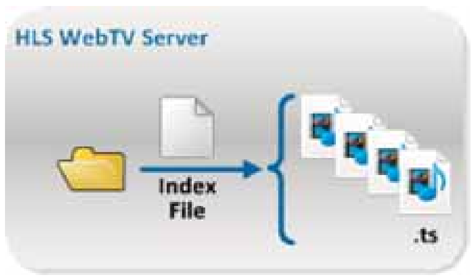

Fig. 7 HLS webTV Server file organization. The index file indicates where are located the .ts files storing fragments [22]

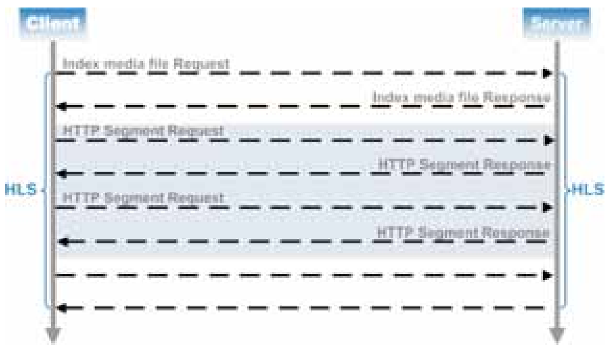

Fig. 8 HLS Communication [22]

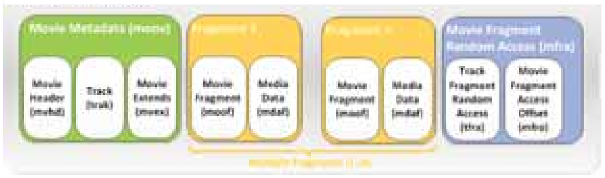

Fig. 9. High level ISO file format structure with multiple fragments used by MS-SSTR [19]

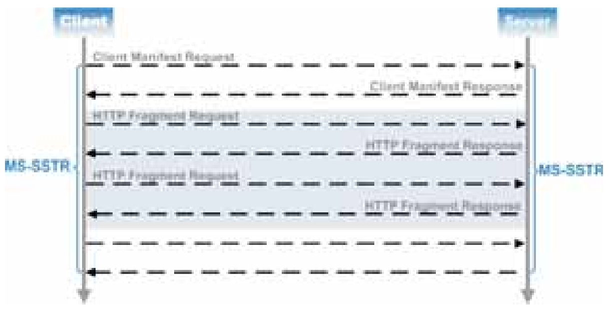

Fig. 10 MS-SSTR Communication [19]

HTTP mainly uses TCP as a transport protocol and is widely used in WebTV. It is a stateless application layer protocol, based on a connection between client and server to provide a communication channel. The basic communication unit is a message; the client sends to the server a HTTP request message and the server answers with the HTTP response message. HTTP also provides a list of methods applied to the message sent. It is important to note the possibility to transfer a message divided into chunks when needed [18].

One main difference between HTTP streaming and other HTTP transactions is that it does not terminate the response to the client by the server due to the constant request of media segments by the client. HTTP streaming uses persistent connection, *Connection: Keep-Alive*, to receive all media segment client requests.

HTTP based progressive download has been further developed into the latest HTTP adaptive streaming. There are multiple options among which are HLS developed by Apple and MS-SSTR by Microsoft. The differences between HTTP based protocols HLS and MS-SSTR are depicted in Table V.

1) HTTP Progressive Download: HTTP progressive download is a hybrid technique between HTTP download and RTP streaming. It downloads the media file in very small parts instead of downloading the entire media file before the client's play-out. To accomplish that, the server stores the media file in small segments, usually from 2sec to 20sec, and an index file with segment transmission, segment information and media resources. The process begins by sending the requested media resource index file to the client. Then streaming via HTTP begins with an HTTP request for the segments and the download. [7] An example of HTTP Progressive Download is HLS provided by Apple and described in subsequent section IV- D.

2) HTTP Adaptive Streaming: Adaptive streaming utilizes HTTP progressive download to accomplish streaming features with added functions such as adapting the media delivery to the client's end device or to Network conditions. Adaptive streaming exploits one of the HTTP characteristics which is to deliver a video/audio file fragmented into small segments or chunks. The media file is stored at the server in multiple small segments or in a unique file virtually fragmented into segments [19].

Segments or chunks are usually 2-4 seconds long and usually begin with an I-frame as well as following GOP boundaries to provide chunk independence from previous or posterior chunks to further facilitate the separated chunk decoding at client-side [19].

A server can store the different versions of the same media file and deliver to a client one version or the other. The same media file can be stored in different quality levels or different bit-rates. The segment stored system allows clients to change the version delivered during streaming, reacting to a client's petitions, the client's end device or to varying network conditions [7].

MS-SSTR and HLS differences Tab. 5

HLS	MS-SSTR
Apple	Microsoft
Uses with a HTTP server	Needs a server-specific IIS extension
One Index file	Client and Server Manifest file
Index files use M3U8 specification	Client's manifest use XML format
H.264 visual codec	H.264 visual codec
MP3 and AAC audio codec	AAC audio codec
Each segment stored in a ts file	One MP4 virtual fragmented file
Media divided into media segment	Media divided into fragments

Protocols characteristics Tab. 6

RTSP	HLS	MS-SSTR	RTMP
IPTV	webTV	webTV	webTV
RTP packets	HTTP segments	HTTP fragments	RTMP chucks
IETF	Apple (IETF)	Microsoft	Adobe Flash
TCP	TCP	TCP	TCP
MP2T	MP2T	MPEG4 part 14	Multiple
Stateful	Stateless	Stateless	Stateless
No handshake	No handshake	No handshake	Handshake connection

Research in adaptive streaming includes the delivery of real-time content such as sports [20]. The latest research in media delivery at mobile platforms uses adaptive streaming to adapt the media delivery to a device's battery consumption.

D. HLS

HLS is an IETF open standard Internet-Draft developed by Apple. Only the latest versions of QuickTime on IPhone 3.0 support the protocol although any HTTP server can use it to deliver the media data. [21] It follows all characteristics of HTTP Progressive Download, i.e., media delivery via HTTP of small fragments, chunks, and the initial media play-out before the complete media download. Finally the HTTP server stores the media source in different bit-rates and one or the other is chosen depending on client or network conditions. In Fig. 7 the HLS server file organization is depicted and the HLS communication process between server and client is described in Fig. 8.

The media container is restricted to MPEG-2 Transport (MP2T) streams. Although in theory there are no media codec restrictions, currently only AAC and MP3 for audio and H.264 for video are accepted. Files are stored at server-side in MP2T files. A single media stream is stored in multiple MP2T files, one for each fragment called media segment files. [22] The index file, also located at server stores the information to locate the MP2T media segment files required to play the entire media stream [21].

The index file follows M3U8 specification which is a play list specification extended from MP3 play list (M3U). [21] Every media is encoded in different bit-rates and has its own index file, as shown in Fig. 7. Index files contain metadata with information about the sequence in which to play the fragments related to a single media source, the location of the MP2T files containing the segments and the alternative media files available in case the request bit-rate is not available [21].

The client requests the media source index file from the server. As a result the server sends it to the client, as shown in Fig. 8. The client in possession of the index file is able to request the fragments in the right order, indicating their server location for the media stream play-out [21].

E. MS-SSTR

MS-SSTR is based on HTTP streaming and designed by Microsoft. It uses adaptive streaming to adapt the media delivery based on local bandwidth and client's CPU. Its main characteristics are its exclusive use of MP4 container, 14496-14 [23], and the use of two index files. First of all the media file uses MP4 format and is structured by splitting the media into smaller fragments within the file and secondly it uses two index files, the server and client manifest [7].

Both manifest index files are stored at server. The server index manifest file provides the server information about the media tracks, bit rates and files, whereas the client index manifest lists the tracks available to the client [7]. In Fig. 9 the ISO file format structure used by MS-SSTR is depicted.

The communication model is based in manifest and fragment request. Assuming an HTTP connection between client and server, the client's first step is to request the Client index manifest which is delivered by the server via the manifest response. After the positive reception of the manifest file, client begins the fragment request to server which sends back the fragment response. This process, fragment request and response, is repeated multiple times until the completed media delivery is terminated [24]. In Fig. 10 the HLS communication process between server and client is described.

The manifest index file is used by the client to request the media fragments based on the timing and bit-rate information. The server, considering network conditions, relates the fragment's request to the corresponding MP4 file where the fragment is extracted and sent as an independent file [7].

One of the main advantages of MPS-SSTR is the use of MP4 file format. The use of this media container facilitates smooth streaming. The media file is stored in an MP4 format which stores all fragments containing the media chunks to deliver in each fragment request.

5. Testbed Development

This protocol review provides a foundation for our experimental research work. We are developing a media synchronisation testbed to evaluate different synchronisation scenarios using media delivered by different methods. A related concept is Hybrid Broadband Broadcast TV (HbbTV) which provides users with IPTV, WebTV and Broadcast TV on a single device. One scenario being evaluated is to synchronise a TV channel delivered via IPTV and a Radio channel delivered via Internet. The TV channel is delivered using RTP and MP2T as indicated in [25]. Regarding the Internet Radio channel, the delivery method can vary. The first step has been to stream to the client via RTP over MP3 and synchronise this mp3 audio with the related video stream. In future work, other options deploying other delivery methods such as HTTP or RTMP will be deployed.

The integration of video and radio streams using RTP provides us with the simplest media delivery method. So far two approaches have been coded, audio substitution and audio addition. The audio substitution replaces the audio from the TV channel and the audio addition creates a new audio channel with the audio from the Internet Radio allowing the user to switch from the original audio and the new audio and vice versa.

We anticipate that such services will be of most potential in the live sporting domain. The media files used in our testbed are related to the same sport event, the Champions League Final FC Barcelona vs. Manchester United year 2011. The video is from SkySportsHD channel with English audio, in MP2T format. The Radio media file, in MP3 format, is the radio transmission of the same event from the Catalan National Radio Station Catalunya Radio.

6. Conclusion

Media traffic over IP Networks is constantly increasing and various protocols are used, each of them suited to particular scenarios. The choice of protocols depends on a range of factors, including, the media platform, IPTV or WebTV, the Transport protocol used, TCP or UDP, IP protocol multicast or unicast, and the server providers' technology.

In this paper we have reviewed the overall suite of media protocols, outlining how they work both individually and in tandem with others to deliver media streams. IPTV systems typically use

RTP and RTSP protocols over UDP and the latest DVB-IPTV standards confirm the future use of these application protocols. Meanwhile WebTV preferences are moving towards HTTP over TCP as a media delivery. The main concern is the multiple HTTP server providers each of which uses their own technology. The principal differences between the protocols are listed in Table VI.

A further complication arises as multiple vendors have developed their own protocols. RTMP, by Adobe Acrobat, HLS by Apple, and MS-SSTR by Microsoft are the main ones. We also briefly outlined our research testbed, which focuses on multimedia synchronisation challenges and opportunities.

Acknowledgement
This Research is sponsored through the Irish Research Council for Science, Engineering & Technology (IRCSET) Enterprise Partnership Program in collaboration with SolanoTech.

References

[1] Cisco Visual Networking Index: Usage. 25th October 2010.
[2] Cisco Visual Networking Index: Forecast and Methodology, 2010–2015, 1st June 2011
[3] MAISONNEUVE, J., DESCHANEL, M., HEILES, J., WEI, L., HONG, L., SHARPE, R., YIYAN, W.: An Overview of IPTV Standards Development, Broadcasting, *IEEE Transactions on,* vol. 55, no. 2, pp. 315–328, 2009.
[4] Internet Engineering Task Force. RFC793, Transmission Control Protocol (TCP). September 1981.
[5] Internet Engineering Task Force. RFC768, User Datagram Protocol (UDP). August 1980.
[6] MA, K. J., BARTOS, R., BHATIA, S., NAIR, R.: Mobile Video Delivery with HTTP, *Communications Magazine,* IEEE , vol. 49, no. 4, pp. 166–175, 2011.
[7] VAN DEURSEN, D., VAN LANCKER, W., VAN DE WALLE, R.: *On Media Delivery Protocols in the Web,* Multimedia and Expo (ICME), 2010 IEEE Intern. Conference on, pp. 1028–1033, 19–23 July 2010.
[8] GOLDBERG, G.: IPTV-ID-0087. *RTP/UDP/MPEG-2 TS as a Means of Transmission for IPTV Streams.* Intern. Telecommunication Union (ITU), Telecommunication Standardization sector, Study Period 2005–2008. Source: Cisco system Inc., USA
[9] BORONAT, S. F., GUERRI, C. J. C., LLORET, M. J.: *An RTP/RTCP Based Approach for Multimedia Group and Inter-stream Synchronisation.* Springer Science + Business Media, LLC, 2008.
[10] BASSO, A., CASH, G. L., CIVANLAR, M. R.: Real-Time MPEG-2 delivery based on RTP: Implementation Issues, *Signal Processing: Image Communication,* No. 15, 1999, pp. 165–178.
[11] MACAULAY, A., FELTS, B., FISHER, Y.: *White Paper IP Streaming of MPEG-4: Native RTP vs. MPEG-2 Transport Stream.* Envivio, October 2005.
[12] BASSO, A., VARAKLIOTIS, S.: *Transport of MPEG-4 over IP/RTP.* Multimedia and Expo, 2000. ICME 2000. 2000 IEEE Intern. Conference on, vol. 2, pp. 1067–1070, 2000.
[13] Internet Engineering Task Force. RFC3550, RTP: A Transport Protocol for Real-Time Applications. July 2003.
[14] Internet Engineering Task Force. RFC2326, Real Time Streaming Protocol. July 1998.
[15] Internet Engineering Task Force. RFC4566, SDP: Session Description Protocol. July 2006.
[16] Real-Time Messaging Protocol (RTMP) Specification 1.0. Adobe Systems Incorporated. April 2009.
[17] CONKLIN, G. J., GREENBAUM, G. S., LILLEVOLD, K. O., LIPPMAN, A. F., REZNIK, Y. A.: Video Coding for Streaming Media Delivery on the Internet, *Circuits and Systems for Video Technology,* IEEE Transactions on, vol. 11, no. 3, pp. 269–281, Mar 2001.
[18] Internet Engineering Task Force. RFC2616, Hypertext Transfer Protocol – HTTP/1.1. June 1999.
[19] ZAMBELLI, A.: *IIS Smooth Streaming Technical Overview.* Microsoft Corporation March 2009
[20] SHIH-FU, CH., DI, ZH., KUMAR, R.; *Real-time Content-based Adaptive Streaming of Sports Videos,* Content-Based Access of Image and Video Libraries, 2001. (CBAIVL 2001). IEEE Workshop on, pp.139–146, 2001.
[21] Informational Internet Draft, Work in progress. HTTP Live Streaming. draft-pantos-http-live-streaming-06. October 2011.
[22] IOS Developer Library. HTTP Live Streaming Overview. Networking & Internet. Apple Inc. April 2011.
[23] ISO/IEC 14496-14. Information technology Coding of audio-visual objects Part 14: MP4 file format. December 2003E.
[24] MS-Smooth Streaming Protocol Specification (MS-SSTR) v20110610. June 2011.
[25] ETSI TS 102 034 V1.4.1 (2009-08). Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks.