

(51) International Patent Classification:
H04L 29/06 (2006.01)(21) International Application Number:
PCT/EP2011/055030(22) International Filing Date:
31 March 2011 (31.03.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10305336.9 1 April 2010 (01.04.2010) EP(71) Applicant (for all designated States except US): **THOMSON LICENSING** [FR/FR]; 1-5 rue Jeanne d'Arc, F-92130 Issy-les-Moulineaux (FR).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **BICHOT, Guillaume** [FR/FR]; c/o Technicolor R&d France, 1 Avenue Belle Fontaine, F-35576 Cesson Sevigne Cedex (FR). **GOUACHE, Stéphane** [FR/FR]; c/o Technicolor R&d France, 1 Avenue Belle Fontaine, F-35576 Cesson Sevigne Cedex (FR).(74) Agents: **PERROT, Sébastien** et al.; 1-5 Rue Jeanne D'arc, F-92130 Issy Les Moulineaux Cedex (FR).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))

Published:

— with international search report (Art. 21(3))

(54) Title: A METHOD FOR RECOVERING CONTENT STREAMED INTO CHUNK

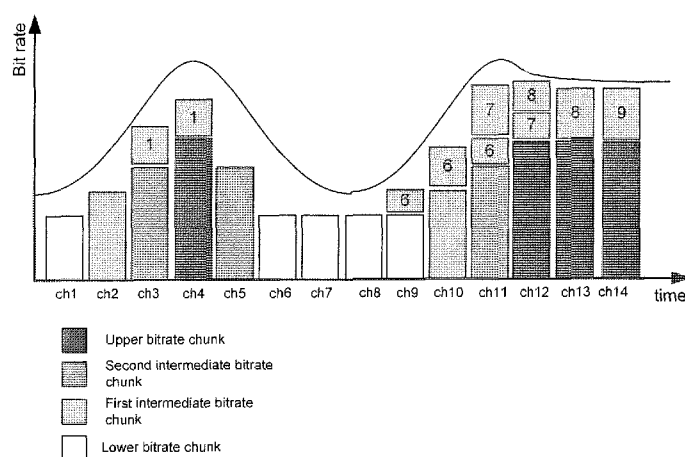


FIG. 4

(57) Abstract: The present invention concerns an adaptive streaming method, at a client device, for receiving content split into chunks corresponding to a content duration, the chunks being encoded at a server into at least a first and a second formats, the first format corresponding to a better content rendering quality level than the second format, a chunk being received during a chunk reception period. According to the invention, the method comprises the steps of measuring the available bandwidth between the client and the server for the next chunk reception period, requesting the server to send a chunk encoded at a format so that the chunk can be received during the next chunk reception period, and if some bandwidth is available for the next chunk reception period, requesting the server to send a part of a chunk encoded at a second format, the chunk having been received encoded at the first format.

A method for recovering content streamed into chunk

FIELD OF THE INVENTION

The present invention relates generally to adaptive video streaming and in particular to a method for recovering the adaptive video
5 streaming content.

BACKGROUND OF THE INVENTION

This section is intended to introduce the reader to various aspects of art, which may be related to various aspects of the present invention that
10 are described and/or claimed below. This discussion is believed to be helpful in providing the reader with background information to facilitate a better understanding of the various aspects of the present invention. Accordingly, it should be understood that these statements are to be read in this light, and not as admissions of prior art.

15 Media delivery streaming solutions are mainly based on protocols such as Real time streaming Protocol (RTSP) as defined in the IETF RFC 2326, Microsoft Media Server (MMS) proprietary protocol from Microsoft or Real Time Messaging Protocol (RTMP) proprietary protocol from Adobe Systems. More recently new streaming techniques based on HTTP protocol
20 have emerged. Adaptive streaming technology provides a way to compensate erratic network behavior regarding available bandwidth by continuously and gracefully upgrading or downgrading the video quality in order to fit with the bandwidth constraint. More precisely, a video stream is encoded into several encoded bit streams, each corresponding to a bit rate
25 constraint, such as for example 300 kbps, 600 kbps, 1200 kbps 2000 kbps or 3000 kbps. Each of these streams is then split into chunks, representing for example two seconds duration, all well aligned in a way that each chunk starts with a reference frame; no frame of any given chunk references a frame in another chunk. In other words, the video stream is cut into short
30 segments called chunks and encoded to the desired delivery format. Chunks are typically 2-to-4-seconds long. At the video codec level, this typically means that each chunk is cut along video Group of Pictures (GOP)

boundaries and has no dependencies on past or future chunks and GOPs. This allows each chunk to later be decoded independently of other chunks.

A client device requests a HTTP server, to send a chunk attached to a particular bit rate regarding the estimated bandwidth, where the available
5 bandwidth can be measured e.g. by measuring the round trip time used by of an HTTP request/response. The video stream is then delivered, chunk after chunk based on client requests. This is illustrated in figure 1, which shows four levels of chunk bit rates, from the lower to the upper bit rate. Video chunks, all corresponding to a fixed duration, are more or less large. A large
10 chunk requires higher bandwidth and provides better video quality. The chunk selection depends on the estimated available bandwidth that corresponds to the curve. Of course, depending on the implementation, the environment, the network technology and the client application, the strategy may be more or less conservative. When conservative, the client requests
15 higher bit rate chunk only after a certain time, ensuring smooth upgrade transition. A less conservative strategy would be to request higher bit rate chunk as soon as the client detects that more bandwidth is available. And in general, the client requests lower bit rate chunk as soon as it detects a bandwidth decrease implying rapid downgrade transition.

20

Examples of such streaming techniques are the "Move Adaptive Stream" from Move Networks, the "HTTP Live Streaming" from Apple Inc. and the "IIS (Internet Information Services) Smooth Streaming" from Microsoft. The benefit of using the HTTP protocol in these streaming
25 solutions is its capability to cross over NAT and firewall seamlessly. These HTTP streaming technologies provide a way to compensate erratic network behavior regarding available bandwidth by continuously and gracefully upgrading or downgrading the video quality in order to fit with the bandwidth constraint.

30 In more details, WO 2005/109224 A2 from Move Networks describes a mechanism in an Agent Controller Module hosted in the client side and able to adapt to the fluctuant network bandwidth thanks to the fact

that the media to stream is previously organized in a plurality of streamlets, also called chunks, each of them being encoded from low to high bit rates. According to the available network bandwidth and some other additional information a monitoring tool incorporated in the Agent Controller Module
5 uses the HTTP protocol to request the server to send the best suited chunk to be streamed over a TCP/IP connection. On an elementary chunk basis the quality is up-shifted or down-shifted according to the Agent Controller Module.

The IIS Smooth Streaming Technical Overview, from Alex
10 Zambelli, Microsoft Corporation, March 2009, describes the IIS Smooth Streaming Technique based on HTTP protocol over TCP/IP connection. The media to be streamed by the Server is previously chopped into chunks representing for example one to ten seconds duration. Then these chunks are encoded according to the H.264/MPEG-4 AVC standard at different bit
15 rates and stored within a MP4 file format container. The mechanism selecting the bit rate according to the network bandwidth fluctuation and requesting seamlessly the corresponding chunks to the server is entirely implemented in the client side through an application code, the Silverlight application. The HTTP Live Streaming supports dynamic switching between streams of
20 different data rates in response to changing connection speeds.

Apple Inc. submitted an Internet Draft to IETF in October 2009 on an HTTP streaming method specification entitled "HTTP Live Streaming draft-pantos-http-live-streaming-02". The HTTP streaming architecture is based on 3 pillars: a Server, a Distribution, through a Web server or a Web
25 caching system, and a Client. The media to be streamed is a video encoded in H.264 and an audio encoded in AAC. At a server, it is encapsulated in MPEG-TS containers and fragmented into chunks of equal duration with a specific tool named Apple stream segmenter. This tool generates the chunks that are saved into *.ts files and an index file *.m3u8 constituting the chunks
30 playlist. Then a client fetches the index file first thanks to an URL pointer. The index file in turn specifies the location of the available media files, decryption

keys, and any alternate streams available. For the selected stream, the client downloads each available media file in sequence.

5 With such adaptive streaming techniques, the video quality is irregular. A stream acquired by a client chunk after chunk does not have a consistent quality as it is a mix of different bit rate chunks. That video may be stored at the receiver to be replayed later on. Usually with such streaming applications, there is no need to resend the stream over the network as the chunks could have been recorded in a local storage facility. However the
10 series of recorded chunks correspond to the network conditions that were experienced during the streaming session resulting in the non consistent quality of the video.

A solution to this problem would be to reacquire from the server the chunks corresponding to the low quality chunks. This can be done as a
15 grouped download before the recorded stream is replayed. However this may be time consuming. Another solution is to replace the low quality chunks on-the-fly during the recorded stream playback. There is no guaranty however that the client would get better quality chunks at the time they are needed either. And, both solutions oblige the client to be connected to the server.

20

SUMMARY OF THE INVENTION

The present invention attempts to remedy at least some of the concerns connected with the low quality chunks in the prior art, by providing methods that opportunistically enhances the chunks quality.

25 To this end, the invention relates to an adaptive streaming method, at a client device, for receiving content split into chunks corresponding to a content duration, the chunks being encoded at a server into at least a first and a second formats, the first format corresponding to a better content rendering quality level than the second format, a chunk being
30 received during a chunk reception period.

According to the invention, the method comprises the steps of measuring the available bandwidth between the client and the server for the

next chunk reception period, requesting the server to send a chunk encoded at a format so that the chunk can be received during the next chunk reception period, and if some bandwidth is available for the next chunk reception period, requesting the server to send a part of a chunk encoded at a second
5 format, the chunk having been received encoded at the first format.

According to an embodiment the method comprises the steps of receiving the chunks and transmitting the chunks to a rendering device.

According to an embodiment the method comprises the step of storing the chunks in a memory.

10 According to an embodiment of the invention, the method comprises the step of using the available bandwidth in successive chunk reception periods for requesting the whole parts of the chunk encoded at the second format.

According to an embodiment of the invention, the method
15 comprises the step of, on reception of the whole parts of a chunk encoded at a second format, swapping in the memory the chunk encoded at the first format with the chunk encoded at a second format.

The invention also relates to an adaptive streaming method, at a
20 client device, for receiving content split into chunks corresponding to a content duration, the chunks being encoded at a server according to the scalable video coding, SVC, technique into a base layer and at least one enhancement layer, a chunk being received during a chunk reception period. To this end the method comprises the steps of measuring the available
25 bandwidth between the client and the server for the next chunk reception period, requesting the server to send a chunk encoded with at least one SVC layer, so that the chunk can be received during the next chunk reception period, and if some bandwidth is available for the next chunk reception period, requesting the server to send at least one enhancement layer of a
30 chunk that have previously been received without said at least one enhancement layer.

According to an embodiment of the invention, the method comprises the step of receiving the chunk and transmitting the chunk to a rendering device.

According to an embodiment of the invention, the method
5 comprises the step of storing the chunks in a memory.

Another object of the invention is a computer program product comprising program code instructions for executing the steps of the method according to the invention, when that program is executed on a computer. By
10 "computer program product", it is meant a computer program support, which may consist not only in a storing space containing the program, such as a computer memory, but also in a signal, such as an electrical or optical signal.

Certain aspects commensurate in scope with the disclosed
15 embodiments are set forth below. It should be understood that these aspects are presented merely to provide the reader with a brief summary of certain forms the invention might take and that these aspects are not intended to limit the scope of the invention. Indeed, the invention may encompass a variety of aspects that may not be set forth below.

20

BRIEF DESCRIPTION OF THE DRAWINGS

The invention will be better understood and illustrated by means of the following embodiment and execution examples, in no way limitative, with reference to the appended figures on which:

- 25
- Figure 1 is an illustration of the delivery of chunks along time;
 - Figure 2 is a block diagram of a system according to the embodiments;
 - Figure 3 is a block diagram of a client device according to the embodiments;
- 30
- Figure 4 illustrates a chunk delivery according to a first embodiment; and

- Figure 5 illustrates a chunk delivery according to a second embodiment.

In Figure 3, the represented blocks are purely functional entities, which do not necessarily correspond to physically separate entities. Namely, they could be developed in the form of hardware or software, or be implemented in one or several integrated circuits.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

It is to be understood that the figures and descriptions of the present invention have been simplified to illustrate elements that are relevant for a clear understanding of the present invention, while eliminating, for purposes of clarity, many other elements found in typical digital multimedia content delivery methods and systems. However, because such elements are well known in the art, a detailed discussion of such elements is not provided herein. The disclosure herein is directed to all such variations and modifications known to those skilled in the art.

Two embodiments are described hereinafter. In the first embodiment content is encoded in several bit rates, while in the second embodiment, content is encoded using scalable video coding. The system according to the embodiments is represented in figure 2. It comprises a client 1 and a server 3 connected through the Internet. The Video File Splitter 5 generates compressed video and audio into chunks. The client is also connected to a player 4. On the server side, chunks are streamed upon the client request, using HTTP protocol over a TCP/IP connection. The client requests chunks according to a method described hereinafter and based on network bandwidth estimation and notably the extra bandwidth.

The client according to the embodiments is illustrated in figure 3. The client comprises a first interface 14 to a network and communicating means 13, which comprises the protocol stacks to communicate to a server

located on the network. In particular the network is the Internet and the communicating means is the TCP/IP stack well known in the art. Of course it could be any other type of network and / or communicating means enabling a client to communicate to a server.

5 The client also comprises a second interface 16 to connect to a video player that is adapted to decode and render the content. Of course the second interface could enable connecting more than one player. The second interface could be an interface to a network enabling to connect one or more players. The client also comprises a processor 11 for processing the
10 applications stored in the client. It comprises a buffer 12 for buffering the chunks received from the server before they are transmitted to the SVC player. The client also comprises a memory 17 to which the chunks received from the server are transmitted. This is preferably a non-volatile memory.

 The client also comprises a non-volatile memory, not represented,
15 for storing applications running at the client. A chunk selector 15 is an application adapted to perform the chunk selection as described hereinafter. The client could be implemented in a gateway device. Alternatively the client comprises an embedded SVC player. It could be then a device such as a set-top box.

20 Alternatively, the memory could alternatively be located in a storing device connected to the client device. The method as described hereinafter is then triggered only if the storing device is connected to the client. This may be triggered automatically on detection of the connected storing device. This may also be triggered on request from the end user.

25 The method at the client for receiving a chunk is summarized as follows:

- the chunk selector selects a chunk according to the method described herein below,
- 30 - the communicating means send a request to the server for receiving the selected chunk,
- the communicating means receive the chunk,

- the chunk is buffered,
- the chunk is sent to the video player.

According to the embodiments, the chunks are also sent to the
5 memory 17. In order to enhance the quality of these stored chunks, the client
uses the remaining available bandwidth to request higher quality chunks as
indicated herein below. The higher quality chunks are then stored in the
memory 17.

10 The method according to the first embodiment is illustrated in
figure 4. A chunk duration corresponds to the video duration attached to the
chunk. The client requests a chunk corresponding to a certain bit rate at
every chunk duration period. The client receives chunks ch1 to ch14. Herein
below the chunk duration is set to two seconds. In a conservative approach,
15 the client requests higher bit rate chunk only after a certain time ensuring
smooth upgrade transition. When the client estimates the bandwidth
available for the next chunk, following a conservative approach, it requests a
chunk corresponding to a bit rate that leaves sufficient bandwidth to request
within the same period of time part of the chunk to be upgraded.

20 Figure 4 shows four levels of chunk bit rates, from the lower to the
upper bit rate, as illustrated in figure 1. For simplicity the chunks
corresponding to a same bit rate have the same size, i.e. all the chunks of a
given bit rate are represented with the same acreage. As illustrated in figure
4, the client has requested the chunk number 3 associated with a bit rate that
25 gives the opportunity to download simultaneously more information. The
client uses the available bandwidth to request a fragment of a first
intermediate bit rate version of chunk 1 which had been initially received as a
low bit rate chunk, illustrated with the block having an index set to one. The
same occurs when requesting the chunk 4. At this time, the client got the last
30 fragment of the first intermediate bit rate version of chunk 1. Further, chunks
6, 7, 8 and 9 are also progressively replaced by their respective first

complete intermediate bit rate versions. Of course, a chunk is replaced only when the corresponding upgraded chunk has been fully received.

Alternatively, other methods are used for upgrading the overall
5 quality of the video locally cached in the client's host. The example shown in figure 4 corresponds to a method that provides an average quality enhancement wherein priority is given to upgrade lowest bit rate chunks first. Another method might prioritize the upgrade of particular chunks first as for instance chunks containing high motion sequence, where low motion
10 sequence can be delivered in low bit rate without too much impact on quality or user experience whereas high motion sequence requires higher bit rate for a consistent quality.

In the system of the first embodiment, the Video File Splitter
15 prepares chunks encoded according to the H.264/MPEG-4 AVC standard at different formats and stored within a MP4 file format container; each format corresponds to a bit rate that is among 300, 600, 1000 or 2000 kbps. The Video File Splitter multiplexes the chunks for producing a MPEG Transport Stream chunk series per supported bit rate. All of these chunks are stored in
20 the HTTP server. A manifest file is sent to the client containing the list of supported bit rates, number of chunks and for each, its size.

The client can then start to request chunks delivery. The client sends one request every 2 seconds, which corresponds to the duration of the video chunk. Practically the client estimates the bandwidth and requests the
25 chunk corresponding to the estimated bandwidth minus a conservative provision. Accordingly, the response, received from the HTTP server arrives after a certain time that is directly linked with the chunk bit rate. Once the response is received, the client computes the remaining time left before requesting the next chunk, the Time left. Then it obtains the number of bytes
30 it can request associated to the previous chunks it wants to upgrade, the BytesMaxNum.

$$\text{Time left} = 2s - \text{chunk round trip time}$$

$\text{BytesMaxNum} = \text{estimated bandwidth (in byte/second)} * \text{time left (in second)}$

Then, the client sends other HTTP request(s) for downloading a certain byte range from the chunk(s) to be upgraded. HTTP request format
5 supports the demand of a certain byte range of a file. However, this requires all HTTP request/response transactions to be serialized. If an HTTP transaction is longer than estimated, this may delay the next chunk request with the risk of starving the player.

Alternatively, the client sets up a second TCP connection. The
10 HTTP requests associated with the process of upgrading the chunks are transmitted over the second TCP connection. The client follows the same process as described herein above. The client estimates the bandwidth and requests the chunk corresponding to the estimated bandwidth minus a conservative provision through the main TCP connection. According to the
15 provisioned estimate, the client computes the number of bytes it can request associated to the previous chunk(s) it wants to upgrade:

$\text{BytesMaxNum} = (\text{estimated bandwidth} - \text{chunk bit rate}) * 2s$

The client may chain in parallel to the main connection, through
20 the second TCP connection, possibly several requests asking for bytes corresponding to fragments of the chunks to be upgraded.

The client may shutdown the second TCP connection independently from the main connection. This may occur if it experiences a bandwidth reduction and wants to give priority to the main connection to
25 maintain a smooth playback experience.

In a second embodiment of the invention, the chunks are encoded according to the Scalable Video Coding (SVC) compression technique that is
30 standardized in H.264/MPEG-4 AVC Annex G. SVC defines three granularity parameters: temporal scalability, spatial scalability and Signal to Noise Ratio (SNR) scalability. According to criteria such as the diversity of targets or

bandwidth overhead, the encoding of a stream does not necessarily require encoding in all the existing types of scalability. This conditions the number of layers for each type of scalability. For HTTP streaming, SVC content is stored in files. SVC files are formatted in either MPEG-2 Transport Streams or MP4 files. These formats allow, among others, to manage the timing information for rendering.

An MPEG2 Transport Stream is usually generated to broadcast a TV channel without any additional data. The broadcast of a TV channel with a SVC video is specified in the standard "Information Technology – Generic Coding of Moving Pictures and Audio: Systems: Transport of scalable video over ITU-T Rec. H.222.0 | ISO/IEC 13818-1; Amendment 3; 03/2009". It defines a Program Map Table (PMT) for the regular TV channel C (one Elementary stream (ES) AVC and one ES audio), with an ES for each enhancement layer "l".

The MP4 file format container is specified in ISO/IEC FDIS 14496-14:2003(E) on "Information technology — Coding of Audio, Picture, Multimedia and Hypermedia Information — Part 14: MP4 file format". It is adapted to SVC content as specified in ISO/IEC 14496-15:2004/FDAM 2:2008(E) on "Information technology — Coding of audio-visual objects — Part 15: Advanced Video Coding (AVC) file format, AMENDMENT 2: File format support for Scalable Video Coding". The SVC file format description, illustrated by examples, can be found in the paper: "File Format for Scalable Video Coding: P. Amon; T. Rathgen; D. Singer; IEEE Transactions on circuits & Systems for video Technology, vol. 17, September 2007".

In the second embodiment, the SVC encoded video is split into chunks. A chunk doesn't only contain one SVC sample but it is a segment of a single contiguous stream, i.e. a collection of consecutive pictures. Hence a chunk has a start time, referred as "chunk offset" hereinafter, and a duration, referred as "chunk duration" hereinafter. Moreover, a chunk is aligned on a group of pictures (GOP) or an integer number of GOPs. Typically the chunk size ranges from 1 to 10s duration. That means each chunk starts with a key frame. The content of a chunk corresponds also to only one layer of a kind of

scalability. All the chunk files have the same format. Each chunk has a unique Uniform Resource Identifier (URI).

Whatever the storage format (MPEG2-TS or MP4 file), the SVC content can be either split into a file per chunk or stored as it is on the server.

5 In the first case, the client requests for an URI while in the second one, it requests for either a timecode and a duration or a byte offset (offset from the beginning of the file) and a byte range. The available chunks are listed in a playlist generated and provided by the streaming server. This playlist can point to other playlists such as one per type of format. Playlists describe the
10 chunk content, such as the codec or the required bandwidth, and the way to request them. The playlist format can be the one described in "HTTP Live Streaming draft-pantos- http-live-streaming-02", extended with codec information to describe the various SVC layers.

The system of the second embodiment is similar to the one of the
15 first embodiment, but content is encoded here into SVC layers instead of being encoded into different quality levels. In particular, the Video File Splitter prepares chunks using four complementary layers; one base layer and three enhancement layers. In the first embodiment, the higher quality chunks are sent to the memory. In the second embodiment, all the chunks are sent to the
20 memory.

The method according to the second embodiment is illustrated in figure 5. The client receives chunks ch1 to ch14, represented by the base layer and enhancement layers not identified by an index. The recovered chunks are represented with an index. Here, the client has requested chunk
25 3 associated with a bit rate that gives the opportunity to download simultaneously more information. The client has used the available bandwidth to request the first enhancement layer piece of chunk 1 which had been initially received as a base layer chunk. The same occurs when requesting the chunk 4; the second enhancement layers of chunks 1 and 2 are requested. With chunk 5, the client requests the third enhancement layer
30 of chunk 1. The same occurs with the chunk 6 where the third enhancement layer of chunk 2 is requested. At the time of chunk 6, the client got the entire

sequence "upgraded" with enhancement layers 1 and 2, and some chunks further improved with enhancement layer 3. Further in time, chunks 3, 5, 6, 7, 8 are also upgraded. For simplicity the base layer and enhancement layers chunks are represented with a same respective size.

5 The second embodiment show a particular arrangement of SVC layers where the enhancement layer 1 depends on the base layer, the enhancement layer 2 depends on the enhancement layer 1 and so on. Alternatively, another SVC scheme may be used where all enhancement layers depend on the base layer. This is of course less flexible but generates
10 less overhead.

References disclosed in the description, the claims and the drawings may be provided independently or in any appropriate combination. Features may, where appropriate, be implemented in hardware, software, or
15 a combination of the two.

Reference herein to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment can be included in at least one implementation of the invention. The appearances of the phrase "in one
20 embodiment" in various places in the specification are not necessarily all referring to the same embodiment, nor are separate or alternative embodiments necessarily mutually exclusive of other embodiments.

Reference numerals appearing in the claims are by way of illustration only and shall have no limiting effect on the scope of the claims.

25

CLAIMS

1. An adaptive streaming method, at a client device, for receiving
5 content split into chunks corresponding to a content duration, said chunks
being encoded at a server into at least a first and a second format, said
second format corresponding to a better content rendering quality level than
the first format, a chunk being received during a chunk reception period,
said method comprising, at the client device, the steps of:
- 10 - measuring the available bandwidth between the client and the
server for a next chunk reception period,
 - requesting the server to send a chunk encoded at any one of
the at least first or second format so that said chunk can be received during
said next chunk reception period, and
- 15 - if said chunk has been received encoded at the first format and
if some bandwidth is available for the next chunk reception period, requesting
the server to send a part of a chunk encoded at a second format.
2. Method according to claim 1, comprising the steps of:
- 20 - receiving said chunk, and
 - transmitting said chunk to a rendering device.
3. Method according to claim 1 or 2, comprising the step of storing
said chunks in a memory.
- 25
4. Method according to any one of the preceding claims,
comprising the step of using said available bandwidth in successive chunk
reception periods for requesting the whole parts of said chunk encoded at the
second format.
- 30
5. Method according to claim 4, comprising the step of, on
reception of the whole parts of a chunk encoded at a second format,

swapping in said memory the chunk encoded at the first format with said chunk encoded at a second format.

- 5 6. An adaptive streaming method, at a client device, for receiving content split into chunks corresponding to a content duration, said chunks being encoded at a server according to a scalable video coding, SVC, technique into a base layer and at least one enhancement layer, a chunk being received during a chunk reception period, said method comprising the steps, at the client device, of:
- 10 - measuring the available bandwidth between the client and the server for the next chunk reception period,
- requesting the server to send a chunk encoded with at least one SVC layer, so that said chunk can be received during said next chunk reception period, and
- 15 - if some bandwidth is available for the next chunk reception period, requesting the server to send at least one enhancement layer of a chunk that have previously been received without said at least one enhancement layer.
- 20 7. Method according to claim 6, comprising the steps of:
- receiving said chunk, and
- transmitting said chunk to a rendering device.
8. Method according to claim 6 or 7 comprising the step of storing
- 25 said chunks in a memory.
9. Method according to any one of the preceding claims, said chunks being requested through an HTTP request.
- 30 10. Computer program product, characterized in that it comprises program code instructions for executing the steps of the method according to claims 1 to 9 when said program is executed on a computer.

1/5

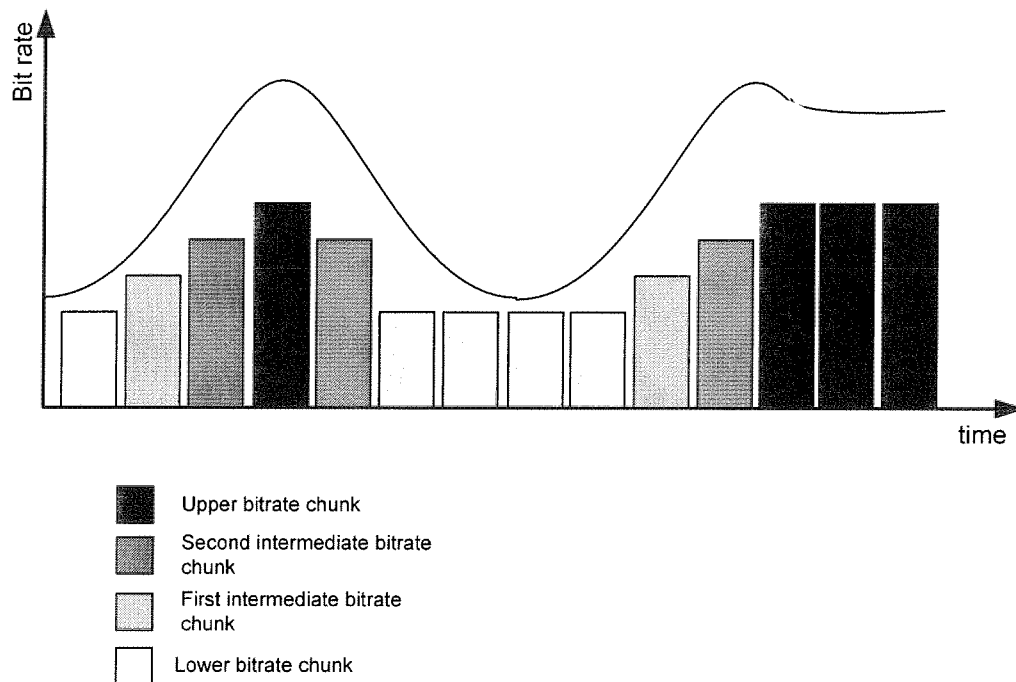


FIG. 1 (PRIOR ART)

2/5

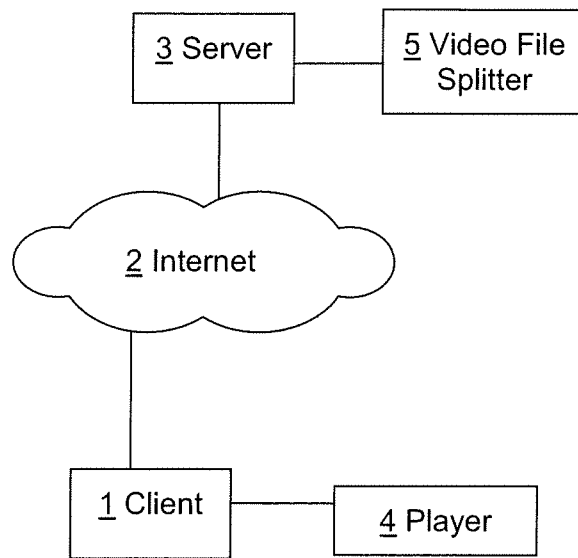


FIG. 2

3/5

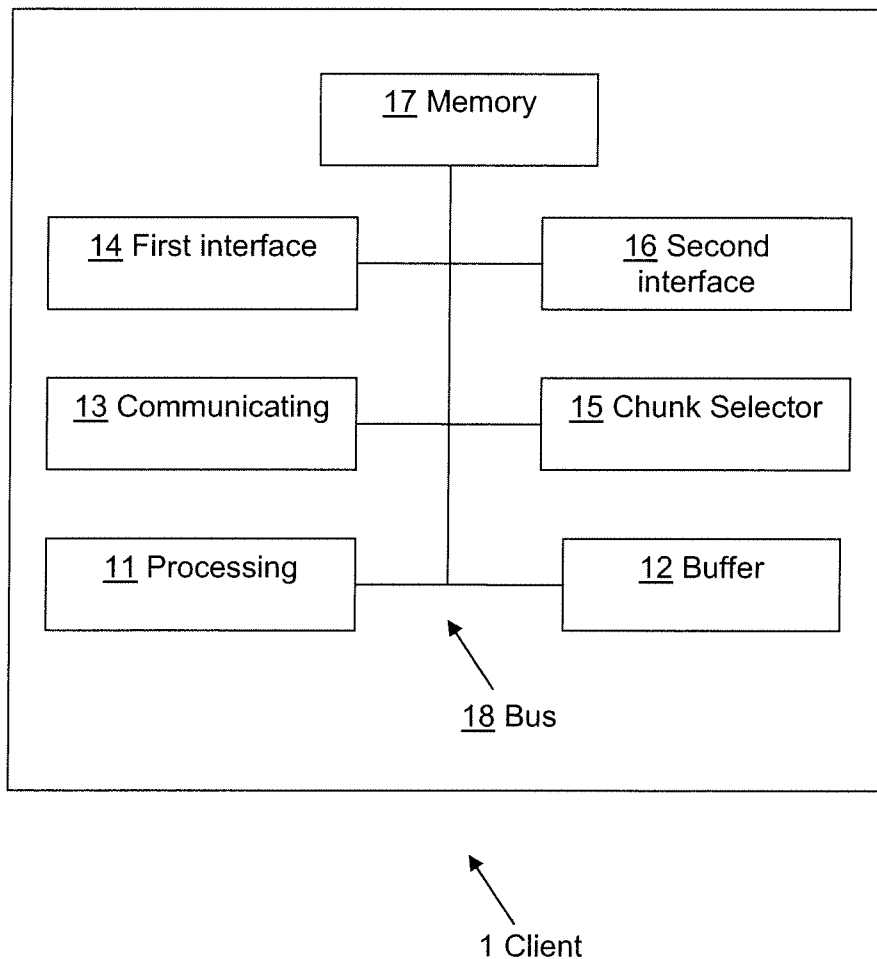


FIG. 3

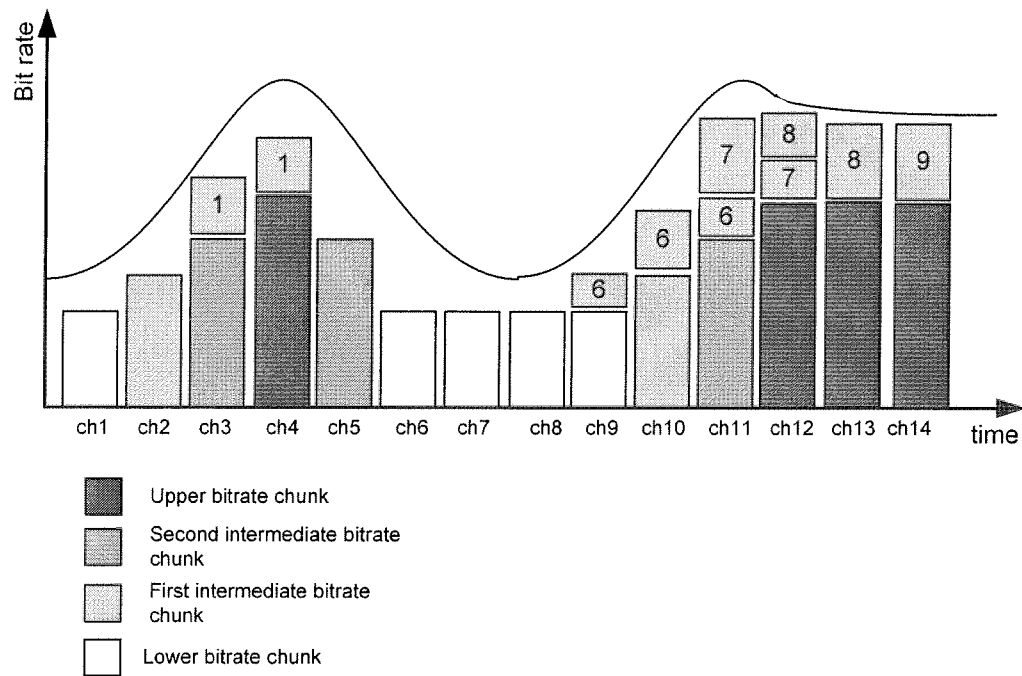


FIG. 4

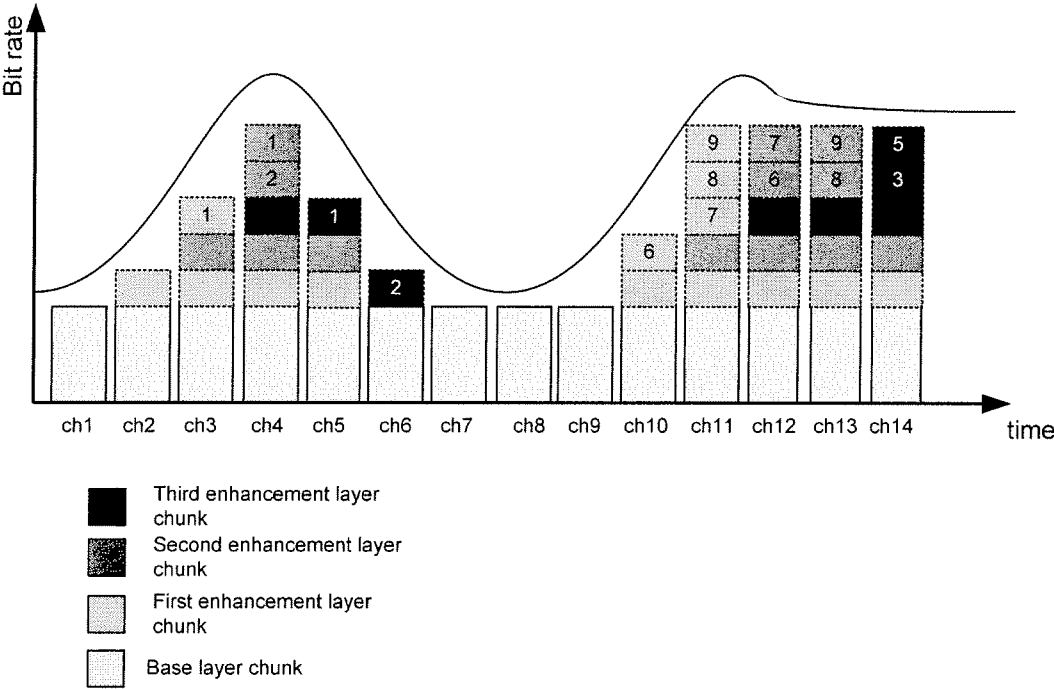


FIG. 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2011/055030

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	SANJEEV MEHROTRA ET AL: "Rate-distortion optimized client side rate control for adaptive media streaming", MULTIMEDIA SIGNAL PROCESSING, 2009. MMSP '09. IEEE INTERNATIONAL WORKSHOP ON, IEEE, PISCATAWAY, NJ, USA, 5 October 2009 (2009-10-05), pages 1-6, XP031550761, ISBN: 978-1-4244-4463-2 page 1 - page 3	1-10
X	US 2003/195977 A1 (LIU TIANMING [CN] ET AL) 16 October 2003 (2003-10-16) paragraph [0005] - paragraph [0030]; figure 4f paragraph [0045] - paragraph [0068] ----- -/-	1-10

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

- "A" document defining the general state of the art which is not considered to be of particular relevance
- "E" earlier document but published on or after the international filing date
- "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- "O" document referring to an oral disclosure, use, exhibition or other means
- "P" document published prior to the international filing date but later than the priority date claimed

- "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
- "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
- "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
- "&" document member of the same patent family

Date of the actual completion of the international search

30 June 2011

Date of mailing of the international search report

13/07/2011

Name and mailing address of the ISA/
European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Jurca, Alexandru

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2011/055030

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2008/294789 A1 (NASSOR ERIC [FR] ET AL) 27 November 2008 (2008-11-27) paragraph [0093] - paragraph [0135] -----	1-10
A	US 2005/262257 A1 (MAJOR R D [US] ET AL) 24 November 2005 (2005-11-24) cited in the application paragraph [0051] - paragraph [0070] -----	1-10
A	ALEX ZAMBELLI: "IIS Smooth Streaming Technical Overview", INTERNET CITATION, [Online] pages 1-18, XP002620446, Retrieved from the Internet: URL: http://www.microsoft.com/downloads/en/details.aspx?FamilyID=03d22583-3ed6-44da-8464-b1b4b5ca7520 > [retrieved on 2011-01-21] cited in the application page 9 - page 16 -----	1-10

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2011/055030

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003195977 A1	16-10-2003	US 2009122878 A1	14-05-2009
		US 2009185618 A1	23-07-2009

US 2008294789 A1	27-11-2008	FR 2916600 A1	28-11-2008

US 2005262257 A1	24-11-2005	AU 2005241481 A1	17-11-2005
		CA 2564861 A1	17-11-2005
		EP 1743249 A2	17-01-2007
		JP 2007535881 A	06-12-2007
		KR 20070061768 A	14-06-2007
		WO 2005109224 A2	17-11-2005
