(12) STANDARD PATENT (11) Application No. **AU 2013251908 B2**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Peer-assisted video distribution

(51) International Patent Classification(s)
H04N 21/2389 (2011.01) ***H04N 21/63*** (2011.01)
H04L 29/06 (2006.01) ***H04N 21/845*** (2011.01)
H04L 29/08 (2006.01) *H04N 21/262* (2011.01)

(21) Application No: **2013251908** (22) Date of Filing: **2013.04.04**

(87) WIPO No: **WO13/160080**

(30) Priority Data

(31) Number	(32) Date	(33) Country
12305464.5	**2012.04.23**	**EP**

(43) Publication Date: **2013.10.31**
(44) Accepted Journal Date: **2017.03.02**

(71) Applicant(s)
THOMSON Licensing

(72) Inventor(s)
Picconi, Fabio

(74) Agent / Attorney
FB Rice, Level 23 44 Market Street, Sydney, NSW, 2000, AU

(56) Related Art
US 2011/0231519

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
31 October 2013 (31.10.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/160080 A1

(51) International Patent Classification:
H04N 21/2389 (2011.01) *H04L 29/08* (2006.01)
H04N 21/845 (2011.01) *H04L 29/06* (2006.01)
H04N 21/63 (2011.01) *H04N 21/262* (2011.01)

(21) International Application Number:
PCT/EP2013/057120

(22) International Filing Date:
4 April 2013 (04.04.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12305464.5 23 April 2012 (23.04.2012) EP

(71) Applicant: **THOMSON LICENSING** [FR/FR]; 1-5 rue Jeanne d'Arc, F-92130 Issy-les-Moulineaux (FR).

(72) Inventor: **PICCONI, Fabio**; 6 rue de la Gaîté, F-75014 Paris (FR).

(74) Agent: **SCHMIDT-UHLIG, Thomas**; Deutsche Thomson OHG, European Patent Operations, Karl-Wiechert-Allee 74, 30625 Hannover (DE).

(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, BN, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ,

[Continued on next page]

(54) **Title**: PEER-ASSISTED VIDEO DISTRIBUTION

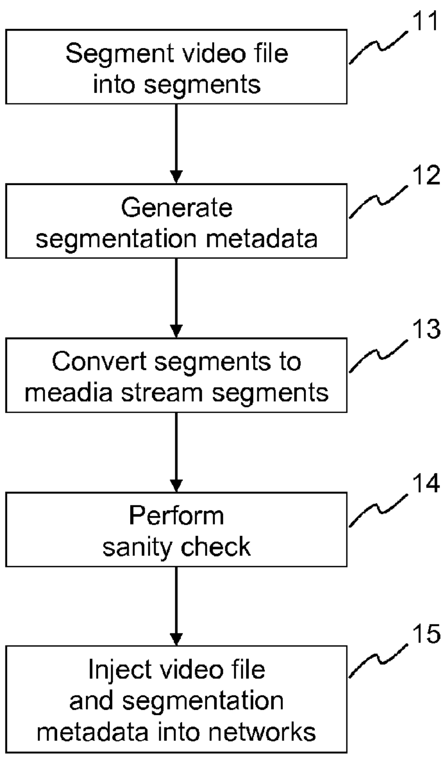
Segment video file into segments
11
Generate segmentation metadata
12
Convert segments to meadia stream segments
13
Perform sanity check
14
Inject video file and segmentation metadata into networks
15

Fig. 3

(57) **Abstract**: A solution for peer-assisted distribution of video content over a broadband network is disclosed. A video file is segmented (11) into segments and segmentation metadata is generated (12). The segments are converted (13) to media stream segments with counter/timestamp consistency. The video file and the segmentation metadata are injected (15) into a peer-assisted video delivery system. The segmentation metadata are then used for initializing download (22) of offsets of the video file that are required for generating (23) segments of the video file at a client.

WO 2013/160080 A1

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).

Published:

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

Peer-assisted Video Distribution

Technical Field

The present invention relates to a solution for peer-assisted distribution of video content over a broadband network, and more specifically to a solution for peer-assisted distribution of video content, which enables a small buffering time until playback, as well as quick fast-forward or rewind operations.

Background

Video accounts for a large fraction of the content consumed today on the Internet. A plurality of websites providing video services constantly deliver video clips or movies to a myriad of different devices, e.g. PCs, TVs, smartphones, and tablets. The consumption of video content is expected to increase as fiber-to-the-home and other high-speed broadband technologies become increasingly widespread.

To support the massive bandwidth requirements of video delivery, service providers employ Content Delivery Networks (CDNs), a set of geographically distributed servers from which users download content. To further improve the efficiency of content distribution, peer-assisted architectures have been proposed, in which users download content from both the CDN and the set-top-boxes or home gateways of other users. In this case, the set-top-boxes or gateways store several video clips on their local hard disk or flash memory and use the available upload bandwidth of the user's home connection to upload fragments of such clips to other users.

For example, the article Wang et al.: "*Is playing-as-downloading feasible in an eMule P2P file sharing system?*", J. Zhejiang Univ.-Sci. C (Comput. & Electron.), Vol. 11 (2010), pp. 465-475, discusses a solution for peer-to-peer (P2P) swarm technologies with media streaming support, which has a playing-

2013251908 05 Jan 2017

as-downloading capability. A new algorithm is introduced, named wish driven chunk distribution (WDCD), which enables the P2P file sharing system to support the video-on-demand (VOD) function while keeping the P2P native downloading speed. A new parameter named next-play-frequency, is added to the content chunk to strike a replication balance between downloading and streaming requests.

The present invention focuses on peer-assisted video distribution systems, and in particular the problem of on-the-fly generation of video segments to support protocols such as HTTP Live Streaming (HLS). The HLS standard describes a protocol for video delivery which supports nice features such as downloading from multiple sources and adaptive stream rates (see: http://tools.ietf.org/html/draft-pantos-http-live-streaming-07). However, one constraint of HLS is that it requires the use of an MPEG Transport Stream (MPEG-TS) container. Therefore, if a video is originally stored in another format, such as MP4 or Matroska (MKV), it must be converted into MPEG-TS before it can be delivered through HLS. Moreover, the MPEG-TS stream must be divided into chunks, or media segments, such that MPEG-TS continuity counters and timestamps are consistent when the segments are stitched back together by the HLS player. The step is known as segmentation. In the following text the terms "segment" and "chunk" will be used indistinctively.

Throughout this specification the word "comprise", or variations such as "comprises" or "comprising", will be understood to imply the inclusion of a stated element, integer or step, or group of elements, integers or steps, but not the exclusion of any other element, integer or step, or group of elements, integers or steps.

2013251908 05 Jan 2017

Any discussion of documents, acts, materials, devices, articles or the like which has been included in the present specification is not to be taken as an admission that any or all of these matters form part of the prior art base or were common general knowledge in the field relevant to the present disclosure as it existed before the priority date of each claim of this application.

Several tools currently exist to perform conversion and segmentation in order to produce HLS-compliant segmentation.

Summary

Existing tools all take as input the entire source file. To see why this is a problem for peer-assisted systems, consider the following. To support a wide range of user devices (PCs, smartphones, etc.), the system can deliver a given video either as a single MP4 file through HTTP, or as multiple MPEG-TS segments through HLS. However, to save storage space, the system only stores each video as an MP4 file. Whenever a user requests the video through HLS, the local peer-to-peer client downloads the MP4 content, converts it into MPEG-TS, and segments the result into HLS chunks. However, since existing HLS segmenter tools work only on entire files, the peer-to-peer client must finish downloading the entire MP4 video before HLS chunks can be created and playback can begin.

It is a feature of the present invention to propose an improved solution for peer-assisted distribution of video content over a network.

According to the invention, a method for distributing a video file comprising frames via a peer-assisted video delivery system comprises the steps of:

- segmenting the video file into temporal segments such that

any frame of the video file is present in one and only one temporal segment, wherein each temporal segment is a self-contained physical object;
- generating segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to the video frames that need to be extracted from the video file to generate the temporal segments, continuity counter values suitable for detecting data loss, and segment start times;
- converting the temporal segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps;
- correcting the continuity counters and/or the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized;
- adding the corrected continuity counters to the segmentation metadata; and
- injecting at least the video file and the segmentation metadata into the peer-assisted video delivery system.

Accordingly, an apparatus for distributing a video file comprising frames via a peer-assisted video delivery system comprises:
- a segmenter configured to segment the video file into temporal segments such that any frame of the video file is present in one and only one temporal segment, wherein each temporal segment is a self-contained physical object; wherein the segmenter is further configured to generate segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to the video frames that need to be extracted from the video file to

generate the temporal segments, continuity counter values suitable for detecting data loss, and segment start times;
- a converter configured to convert the temporal segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps, to correct the continuity counters and/or the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within the media stream segment and between consecutive media stream segments are valid and synchronized, and to add the corrected continuity counters to the segmentation metadata; and
- an injector configured to inject at least the video file and the segmentation metadata into the peer-assisted video delivery system.

According to one aspect of the invention the video file, e.g.an MP4 file or an MKV file, is first segmented into segments. Preferably, these segments are fixed-duration segments. Segmentation metadata are generated during segmentation. The resulting segments are converted to media stream segments, e.g. to MPEG-TS segments, with counter/timestamp consistency. Optionally, a sanity check is performed on the media stream segments. Subsequently, the video file and the segmentation metadata are injected into the peer-assisted video delivery system.

A method for receiving a video file comprising frames from a peer-assisted video delivery system comprises the steps of:
- downloading segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to video frames that need to be extracted from the video file to generate temporal segments of the video file, continuity counter values suitable for

detecting data loss, and segment start times;
- analyzing the segmentation metadata to determine video offsets to be downloaded and to initiate download of the determined video offsets;
- downloading the determined video offsets for at least a full length segment;
- generating video segments from the downloaded video offsets;
- converting the video segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps; and
- correcting the continuity counters using continuity counter values from the segmentation metadata and/or correcting the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized.

Accordingly, an apparatus for receiving a video file comprising frames from a peer-assisted video delivery system comprises:
- a network connector configured to download segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to video frames that need to be extracted from the video file to generate temporal segments of the video file, continuity counter values suitable for detecting data loss, and segment start times, and to download the video offsets for at least a full length segment;
- an analyzer configured to analyze the segmentation metadata to determine video offsets to be downloaded and to initiate download of the determined video offsets;
- a segment generator configured to generate video segments from the downloaded video offsets; and

- a converter configured to convert the video segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps, and to correct the continuity counters using continuity counter values from the segmentation metadata and/or to correct the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized.

For regenerating the media stream segments, in a first step the segmentation metadata file is downloaded. The segmentation metadata are then analyzed to determine which video offsets, e.g. MP4 or MKV offsets, need to be downloaded. Subsequently download of the determined video offsets is initiated. Video segments are then generated from the downloaded video offsets, which are finally converted to media stream segments, e.g. to MPEG-TS segments, with counter/timestamp consistency.

It is an advantage of at least one exemplary embodiment that it allows a peer to produce media stream segments without downloading the entire source video file. This enables a small buffering time until playback, i.e. playback can begin before the entire file is downloaded, as well as quick fast-forward or rewind operations. In addition, as the file is stored in only one format, the solution is very efficient in terms of storage requirements. Furthermore, it is compatible with existing standards.

Brief description of the figures

For a better understanding the invention shall now be explained in more detail in the following description with reference to the figures. It is understood that the invention is not limited to this exemplary embodiment and that specified features can also expediently be combined and/or modified without departing from the scope of the present invention as defined in the appended claims. In the figures:

Fig. 1	illustrates a peer-assisted video delivery system;
Fig. 2	schematically depicts a method according to the invention for peer-assisted video delivery;
Fig. 3	describes segmentation of an MP4 file and generation of a segmentation metadata file;
Fig. 4	shows the regeneration of HLS segments;
Fig. 5	schematically illustrates a device for preparing a video file for injection in a peer-assisted video delivery system; and
Fig. 6	schematically depicts a device for receiving a video file from a peer-assisted video delivery system.

Best mode

In the following description the case of on-the-fly MP4-to-HLS conversion will be considered. However, the invention also works for other input formats, e.g. MKV-to-HLS conversion. In addition, the mechanism behind the invention is not specific to HLS, and can be applied to similar target protocols such as Dynamic Adaptive Streaming over HTTP (DASH).

Fig. 1 illustrates the different components of a peer-assisted system for video delivery via a network 1. It is assumed that a peer-to-peer client runs on each home gateway 2 connected to the network 1. Videos can be watched from different user

2013251908 05 Jan 2017

devices, such as PCs 3 and tablets 4. When a user starts watching a video, the peer-to-peer client on the gateway 2 downloads pieces of video data from other peer-to-peer clients and/or the CDN 5, reassembles the stream, and forwards it to the user device. Of course, the peer-to-peer client may likewise run on the user device instead of the home gateway 2.

Fig. 2 schematically depicts a method according to the invention for peer-assisted video delivery. The method is essentially composed of two steps. In a first step 10 an MP4 file is segmented and a segmentation metadata file is generated. In a second step 20 HLS segments are regenerated.

The first step 10 is executed whenever a MP4 file is first injected into the system, i.e. into the CDN plus the peer-to-peer storage, and becomes available for download. This step analyzes the file and generates some metadata that will be used during the second step 20, which is executed whenever a peer-to-peer client needs to generate a HLS chunk on-the-fly.

Fig. 3 describes the segmentation of the MP4 file and the generation of the segmentation metadata file in more detail. This step is composed of several steps. First the MP4 file is segmented 11 into MP4 segments, preferably fixed-duration segments, and segmentation metadata is generated 12. These segments are converted 13 to MPEG-TS segments with counter/timestamp consistency. Then an MPEG-TS sanity check is performed 14 and the video file and the segmentation metadata are injected 15 into the CDN and the P2P network.

The segmentation step 11 is run on a computer which stores the entire MP4 source file. The computer invokes a segmenter tool to generate all target segments. For instance, if the file is to be segmented into k chunks, then the tool is invoked k

2013251908 05 Jan 2017

times, once for each segment. At each invocation, the tool receives as parameter the segment start and end times. For example, if segments are 5 seconds long, and the third segment is to be generated, then the tool is invoked with "start time t=10" and "end time t=15".

A critical issue here is that the created segments must not contain overlapping video/audio frames, that is, a frame must not be present in more than one segment. Similarly, all frames must be present in at least one segment. These two conditions ensure that stitching the resulting segments will result in continuous and error-free playback.

Today, no existing tools support this type of segmentation out-of-the-box. For instance, the popular tool MP4Box can create individual chunks specified by "start time" and "end time" parameters. However, the resulting chunks will contain overlapping video/audio frames. Nevertheless, a workaround exists for this problem that allows MP4Box to be used without modifying its source code. The latest development version of MP4Box supports the -split-chunk S:E option. This option generates a chunk specified by a start time S and an end time E. However, the tool may adjust the start time to align it to a particular frame. For instance, invoking the tool with "-split-chunk 5:10" may result in a chunk that begins at t=4.96 and ends at t=10. If the previous chunk ends at t=5, then some frames may be present in both chunks, causing problems when the chunks are stitched back together.

A workaround for this problem consists in invoking MP4Box in two passes. During the first pass, it is invoked using the following options (for a duration of 5 seconds):

```
segment 1          -split-chunk 0:5
segment 2          -split-chunk 5:10
segment 3          -split-chunk 10:15
segment 4          -split-chunk 15:20
                              .
                              .
                              .
```

On each invocation, MP4Box may output that it has adjusted the start time to a particular value, for instance t=14.96 instead of t=15. These modified start times are stored in a temporary file.

During the second pass, MP4Box is invoked again to generate all chunks, but replacing the end time of chunks according to the start time of the following chunk. In the above example, this would result in the following options:

```
segment 1          -split-chunk 0:5
segment 2          -split-chunk 5:10
segment 3          -split-chunk 10:14.96
segment 4          -split-chunk 15:20
                              .
                              .
                              .
```

Note that the option for segment 4 specified a start time at t=15. This is intended, as MP4Box will adjust the start time of the chunk to 14.96, exactly as it did in the first pass.

During the second pass, the system monitors the MP4 source file offsets accessed by MP4Box and stores them in the segmentation metadata file. Finally, the adjusted start times are also added to the segmentation metadata file.

The following is an example of a segmentation metadata file that uses JSON encoding (JavaScript Object Notation).

```
[
  {
    "index": 1,
    "start": 0,
    "end": 5.00,
    "mp4offsets": [
          [0,303103],
          [8441856,8444766]
    ],
    "continuity": "…"
  },
  {
    "index": 2,
    "start": 5.00,
    "end": 10.00,
    " mp4offsets ": [
          [0,40959],
          [290816,876543],
          [925696,929791],
          [995328,1011711],
          [8441856,8444766]
    ],
    "continuity": "…"
  },
  {
    "index": 3,
    "start": 10.00,
    "end": 14.96,
    " mp4offsets ": [
          [0,40959],
          [847872,1658879],
            [1687552,1691647],
            [1789952,1810431],
            [8441856,8444766]
      ],
      "continuity": "…"
    },
    {
      "index": 4,
      "begin": 14.96,
      "end": 19.88,
      " mp4offsets ": [
            [0,40959],
            [1626112,2400255],
            [2437120,2441215],
            [2531328,2547711],
            [8441856,8444766]
      ],
      "continuity": "…"
    },
]
```

Note that the MPEG-TS continuity counter values have been omitted in this example file.

One of the insights exploited by the invention is the fact that the segmenter tool will not access the entire source MP4 file when generating a particular segment. For instance, when generating the third segment, the tool will access the beginning or the end of the file, where the MP4 metadata is located, plus a range of offsets which correspond to the video/audio frames that need to be extracted to generate the chunk.

Therefore, according to the invention the segmenter tool is monitored upon each invocation to determine all the source MP4 file offsets that the tool reads. These offsets are saved in the segmentation metadata file and will be used in the regeneration step 20. There are several ways to monitor the source MP4 file offsets read by the tool. One possibility is to invoke the tool using the strace Unix command to trace all file system calls.

During the conversion step 13 all MP4 segments are converted into MPEG-TS containers and stitched together into a single MPEG-TS file. This step 13 must make sure that the stitched MPEG-TS file can be played back without error. This includes making sure that continuity counters are properly incremented and that PCR and PTS/DTS timestamps are valid and synchronized.

One possibility to perform this step is to employ the ffmpeg tool (see http://ffmpeg.org/), which supports conversion from MP4 to MPEG-TS. However, the segments created by ffmpeg cannot be stitched together by simple concatenation, as the values of continuity counters, PCR (Program Clock Reference) and PTS (Presentation Time Stamp)/DTS (Decode Time Stamp) would be incorrect. The solution is to concatenate the MPEG-TS segments produced by ffmpeg, and run a sanitizer tool, which fixes the continuity counters, PCR and PTS/DTS values. This can be achieved as follows.

For the continuity counters, on each segment transition the continuity counter values are modified to be consistent with the last counter value observed in the previous segment. These last counter values are saved in the segmentation metadata file.

2013251908 05 Jan 2017

For the PCR/PTS/DTS values, on each segment transition the PCR/PTS/DTS values will be reset to zero, as each segment is generated individually with a different time reference. However, a simple adjustment suffices. The PCR/PTS/DTS values are increased by a value equal to the start time of the segment, which can be found in the segmentation metadata file.

The MPEG-TS sanity check 14 is optional. It checks the MPEG-TS produced in the conversion step 13 to check that there are no playback errors due to discontinuities or incorrect timestamp values.

The injection step 15 involves injecting the MP4 file into the CDN and peer-to-peer storage spaces. The segmentation metadata file may be stored in the P2P tracker for easy access by peers. The MPEG-TS segments may also be injected into the CDN to avoid on-the-fly conversion on the CDN side.

Fig. 4 shows the regeneration 20 of the HLS segments in more detail. Also this step 20 is composed of several steps. In a first step 21 the segmentation metadata file is downloaded. In a second step 22 the segmentation metadata are analyzed to determine which MP4 offsets need to be downloaded and download of the determined MP4 offsets is initiated. Then MP4 segments are generated 23 and converted 24 to MPEG-TS segments with counter/timestamp consistency.

The regeneration step 20 is executed whenever the video is delivered to the user device using HLS. Typically, the device will request the segments sequentially, i.e. segment 1, segment 2, etc., although segments may be requested out of order after a fast-forward or rewind operation.

2013251908 05 Jan 2017

During the first download step 21 the peer-to-peer client obtains a copy of the segmentation metadata file for that particular video. One possibility is to download it from a tracker, provided the file has been previously been uploaded there.

For the MP4 offset download 22 the peer-to-peer client analyses the metadata to determine which MP4 offsets must be downloaded in order to generate each segment. For instance, in the above given example the offset ranges for the third segment are {[0,40959], [847872,1658879], [1687552,1691647], [1789952,1810431], [8441856,8444766]}. The client then downloads these offsets and writes them to a local MP4 file. As in any peer-assisted system, such download might involve transfers from other peers and/or from the CDN.

Once this step 22 is over, the local MP4 file contains the original MP4 data in all the offsets that will be read by the segmenter tool to generate a particular segment. Other offsets may contain zeros, garbage, or the original MP4 data if those offsets have been previously fetched for other segments.

During the segment generation step 23 the segmenter tool is called to generate the specified segment. If the workaround presented above is employed, then the MP4Box options must be adjusted using the start times contained in the metadata file.

Finally, during the conversion step 24 the MP4 segment is converted into an MPEG-TS one, and sanitized to avoid discontinuities and incorrect timestamp values. This step is similar to the conversion step 13, the only difference being that the continuity counter values are adjusted using the information contained in the segmentation metadata file, as the previous video segment may not be available locally.

A device 30 for distributing a video file via a peer-assisted video delivery system is schematically illustrated in Fig. 5. The device 30 comprises a segmenter 31 for segmenting the video file into segments, preferably fixed-duration segments, and for generating segmentation metadata. A converter 32 converts these segments to MPEG-TS segments with counter/timestamp consistency. An optional sanity checker 33 is provided for performing an MPEG-TS sanity check. Finally, an injector 34 injects the video file and the segmentation metadata into the CDN and the P2P network.

A device 40 for receiving a video file from a peer-assisted video delivery system is schematically depicted in Fig. 6. The device 40 comprises a network connection 41 for downloading the segmentation metadata file. An analyzer 42 analyzes the metadata to determine which video offsets must be downloaded and initiates download of the determined video offsets. A segment generator 43 generates the video segments, which are then converted by a converter 44 to MPEG-TS segment with counter/timestamp consistency.

2013251908 05 Jan 2017

Claims:

1. A method for distributing a video file comprising frames via a peer-assisted video delivery system, the method comprising:
- segmenting the video file into temporal segments such that any frame of the video file is present in one and only one temporal segment, wherein each temporal segment is a self-contained physical object;
wherein the method further comprises:
- generating segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to the video frames that need to be extracted from the video file to generate the temporal segments, continuity counter values suitable for detecting data loss, and segment start times;
- converting the temporal segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps;
- correcting the continuity counters and/or the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized;
- adding the corrected continuity counters to the segmentation metadata; and
- injecting at least the video file and the segmentation metadata into the peer-assisted video delivery system.

2. The method according to claim 1, wherein the video file is segmented into fixed-duration temporal segments.

3. A method for receiving a video file comprising frames from a peer-assisted video delivery system, the method comprising:
- downloading segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to video frames that need to be extracted from the video file to generate temporal segments of the video file, continuity counter values suitable for detecting data loss, and segment start times;
- analyzing the segmentation metadata to determine video offsets to be downloaded and to initiate download of the determined video offsets;
- downloading the determined video offsets for at least a full length segment;
- generating video segments from the downloaded video offsets;
- converting the video segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps; and
- correcting the continuity counters using continuity counter values from the segmentation metadata and/or correcting the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized.

4. The method according to one of the preceding claims, wherein the video file is an MP4 file or an MKV file.

5. The method according to one of the preceding claims, wherein the media stream segments are MPEG transport stream segments.

6. An apparatus for distributing a video file comprising frames via a peer-assisted video delivery system, the apparatus comprising:
- a segmenter configured to segment the video file into temporal segments such that any frame of the video file is present in one and only one temporal segment, wherein each temporal segment is a self-contained physical object; wherein the segmenter is further configured to generate segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to the video frames that need to be extracted from the video file to generate the temporal segments, continuity counter values suitable for detecting data loss, and segment start times;
and in that the apparatus further comprises:
- a converter configured to convert the temporal segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps, to correct the continuity counters and/or the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized, and to add the corrected continuity counters to the segmentation metadata; and
- an injector configured to inject at least the video file and the segmentation metadata into the peer-assisted video delivery system.

7. The apparatus according to claim 6, wherein the segmenter segments the video file into fixed-duration temporal segments.

2013251908 05 Jan 2017

8. An apparatus for receiving a video file comprising frames from a peer-assisted video delivery system, the apparatus comprising:
- a network connector configured to download segmentation metadata comprising at least information about video offsets relative to a beginning of the video file corresponding to video frames that need to be extracted from the video file to generate temporal segments of the video file, continuity counter values suitable for detecting data loss, and segment start times, and to download the video offsets for at least a full length segment;
- an analyzer configured to analyze the segmentation metadata to determine video offsets to be downloaded and to initiate download of the determined video offsets;
- a segment generator configured to generate video segments from the downloaded video offsets; and
- a converter configured to convert the video segments into media stream segments in accordance with a media stream format, which uses continuity counters and/or timestamps, and to correct the continuity counters using continuity counter values from the segmentation metadata and/or to correct the timestamps of the media stream segments such that the continuity counters within each media stream segment and between consecutive media stream segments are properly incremented and/or the timestamps within each media stream segment and between consecutive media stream segments are valid and synchronized.

9. The apparatus according to one of claims 6 to 8, wherein the video file is an MP4 file or an MKV file.

10. The apparatus according to one of claims 6 to 9, wherein the media stream segments are MPEG transport stream segments.

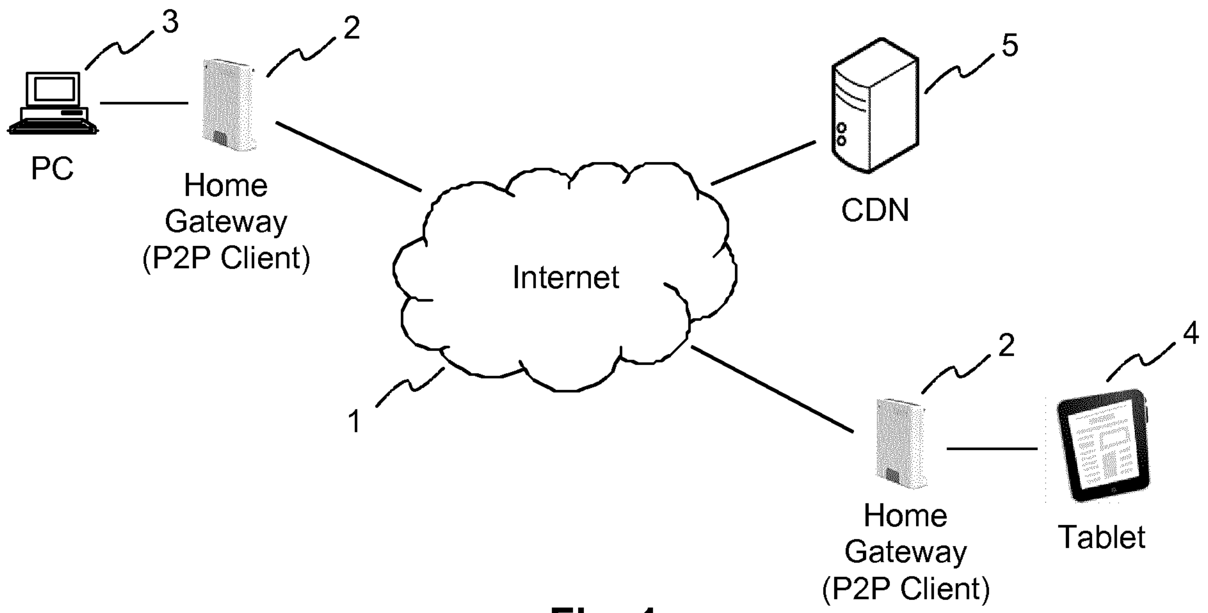
3
2
5
PC
Home Gateway (P2P Client)
CDN
Internet
2
4
1
Home Gateway (P2P Client)
Tablet

Fig. 1

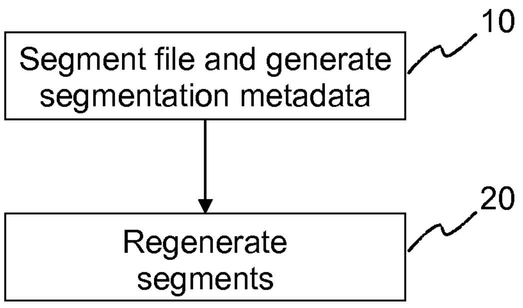
Segment file and generate segmentation metadata
10
Regenerate segments
20

Fig. 2

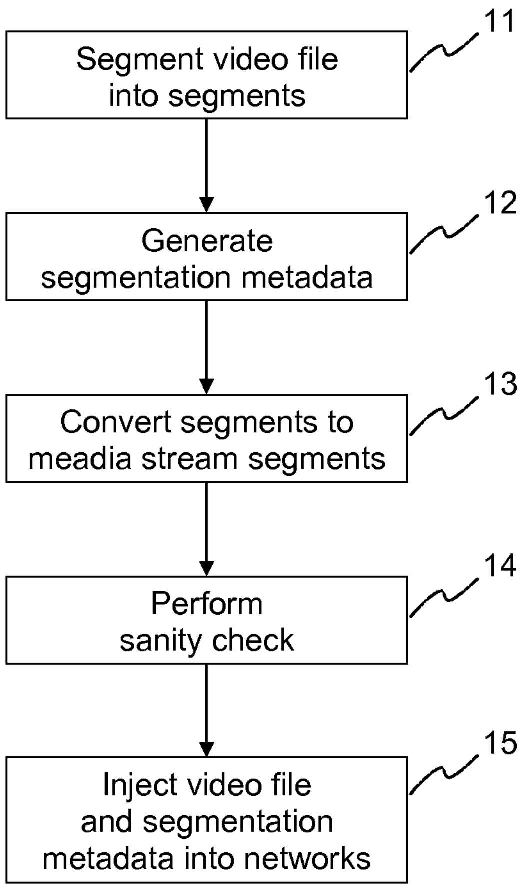
Segment video file into segments
11
Generate segmentation metadata
12
Convert segments to meadia stream segments
13
Perform sanity check
14
Inject video file and segmentation metadata into networks
15

Fig. 3

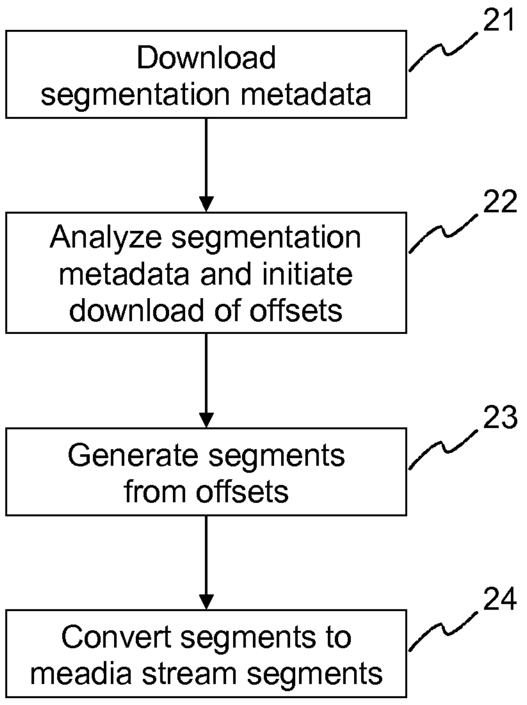
Download segmentation metadata
21
Analyze segmentation metadata and initiate download of offsets
22
Generate segments from offsets
23
Convert segments to meadia stream segments
24

Fig. 4

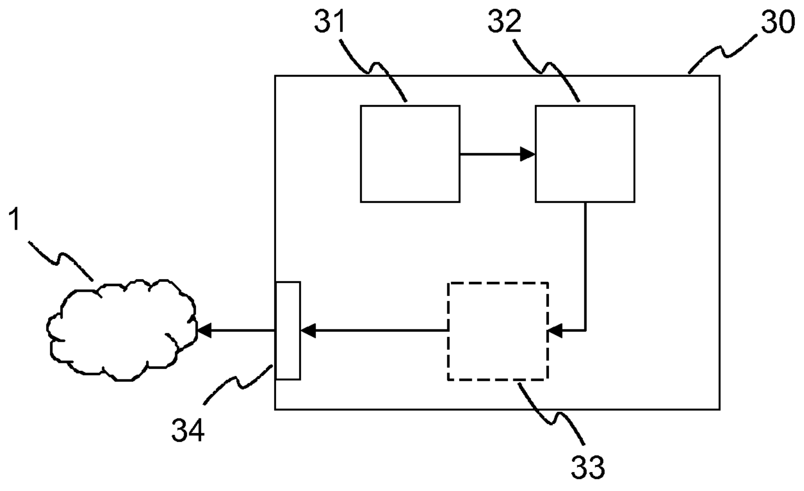

Fig. 5

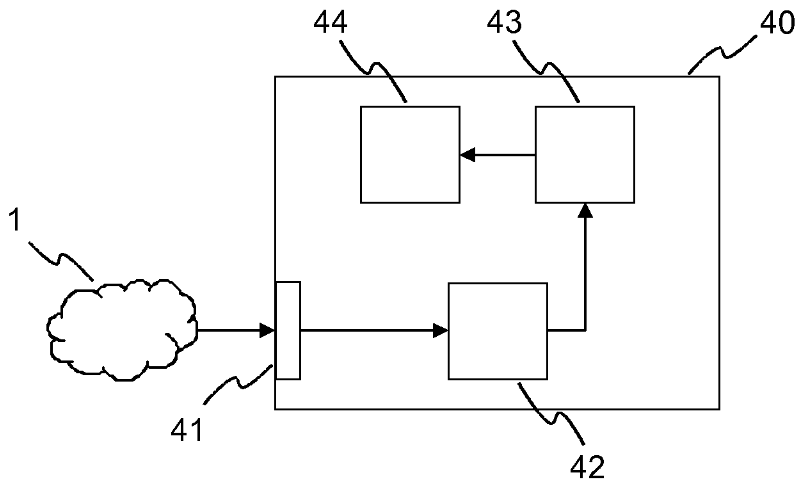

Fig. 6