Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://asr.nsps.org.ng/index.php/asr

African Scientific Reports 2 (2023) 100

African Scientific Reports

Evaluation of an Adaptive Multimedia Streaming in Mobile Cloud Computing for Slow-Speed Networks

O. E. Ojo[a,*], O. A. Oyinloye[b], A. O. Adejimi[a], T. Adejumo[a]

[a]*Department of Computer Science, Federal University of Agriculture Abeokuta, Nigeria.*
[b]*Department of Computing, University of Ilesa, Ilesa, Nigeria.*

Abstract

Multimedia cloud (MC) is an aspect of cloud computing that facilitates the effective use of multimedia services by end users in the context of cloud infrastructures. Despite rising network traffic, cloud computing technology provides novel strategies for disseminating visual content; adaptive encoding is implemented at the cloud server to optimise performance. However, streaming video over the Internet has caused a slew of problems, including sporadic interruptions, delays, inadequate bandwidth, and oscillating link conditions, all of which contribute to poor quality of service (QoS). This study presents an adaptable streaming method to grapple with delays, sporadic interruptions, and bandwidth alterations. The adaption logic concept was used to develop the scheme, which was then put into practice utilising the Java programming language and cloud computing. A variety of network circumstances were used to test the method using pre-recorded video sequences that were separated into chunks with fewer frames. The evaluation findings showed that the suggested streaming technique can dynamically adapt to different bandwidth changes, making it ideal for slow-speed network situations. The system is also capable of delivering seamless, interruption-free video playback.

DOI:10.46481/asr.2023.2.2.100

Keywords: Multimedia cloud, Video Streaming, DASH, Xenplayer

Article History :
Received: 16 March 2023
Received in revised form: 17 May 2023
Accepted for publication: 23 May 2023
Published: 18 June 2023

Communicated by: Tolulope Latunde

1. Introduction

The field of cloud computing is making great strides in the academic community as well as in industry. The long-term goal of this kind of innovation involves offering emulated, dispersed, and elastic resources so that computing can

[*]Corresponding author tel. no:
Email address: ojoeo@funaab.edu.ng (O. E. Ojo)

be fully realised as a utility [1, 2]. With the advent of cloud computing, it is now feasible to create applications that run entirely on the cloud. On the other hand, the cloud provider provides access to an extensive network of computers and related services at a variable cost per user. However, large software system providers create applications for online socialising and shopping. Using cloud infrastructure services could minimise costs and boost the end-user quality of service for these applications [3]. Thus, cloud computing is a distributed and parallel network that is autonomously provided and displayed as one or more homogenous resources for computation based on service-level agreements signed between the cloud service provider and the end user [4, 5]. It is characterised as a model for providing rapid provisioning and release of a shared pool of configurable computing resources such as servers, storage, and networks with minimal intervention from the service provider [6]. The cloud computing paradigm encompasses four distinct deployment models, three distinct service models, and a set of defining characteristics. Cloud computing relies on a multitude of servers housed in data centres, numbering from hundreds to thousands [7]. The diagram depicted in Figure 1 showcases the architecture of a data centre that is based on cloud computing technology. According to reference, [8], virtualization services are responsible for managing storage servers. These services offer measures to ensure security and resilience against failure. The uppermost tier is composed of application services such as social networking, web hosting, and data processing. The application's pattern is contingent upon the quality of service (QoS) and service requisites.

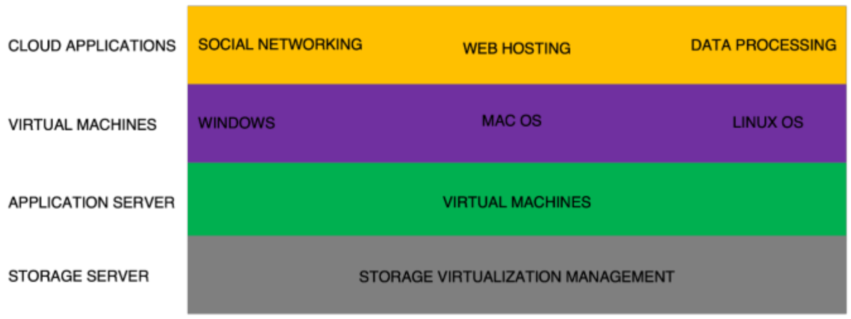


Figure 1. Cloud-based typical data center [8]

1.1. Adaptive multimedia streaming

The rapid development of multimedia applications has garnered significant attention for diverse organisational needs. Notably, video streaming has been increasingly utilised in enterprise services, including teleconferencing, online education, and streaming services. Additionally, video reconnaissance of target fields or specified objects has been employed for military applications. The process of video tracing requires a significant amount of resources and bandwidth, making it challenging to stream videos over networks with limited bandwidth. The utilisation of bandwidth entails a direct expense for the ultimate consumers [9]. Streaming is a prevalent method of transmitting video content over the Internet, and its technological capabilities are consistently advancing. Video streaming enables the dissemination of visual data, the delivery of conferences, the broadcasting of lectures, and the demonstration of procedural instructions. According to study findings, basic interpersonal interactions such as messaging, voice, and video communication will be a dominant force in mobile commerce on 3G wireless systems and beyond [10]. In the context of video delivery using the hypertext transfer protocol (HTTP), it is common practice to divide videos into segments and encode each segment at varying bitrate levels. The alignment of chunks from various bitrate streams facilitates seamless switching to a different bitrate by the video player, if required, at the juncture of chunk boundaries. The user employs bitrate selection and adaptation algorithms to determine the appropriate bitrate levels for upcoming chunks, to provide an optimal QoS [11]. The utilisation of adaptive bit-rate (ABR) streaming is a viable approach to enhancing the dissemination of HTTP-based video content by leveraging the intricacies of network patterns, particularly in wireless networks. The approach centred around HTTP is intrinsically lacking in statefulness. The implementation of ABR has been observed in various media solutions, including Adobe's HTTP dynamic streaming, Apple's HTTP live streaming, and Microsoft's smooth streaming [12]. The ABR framework for a wireless communication system is illustrated in Figure 2. The process of storing video content on a distant server and transmitting it to a handheld device through a mobile network, which encompasses both a backbone network and an access point, is employed. The video content is encoded using Adaptive Bitrate (ABR) technology, which allows for multiple degrees of quality to be available for playback. To enable playback of a clip at a designated resolution, it is initially segmented into several smaller units, each with a brief duration [13].

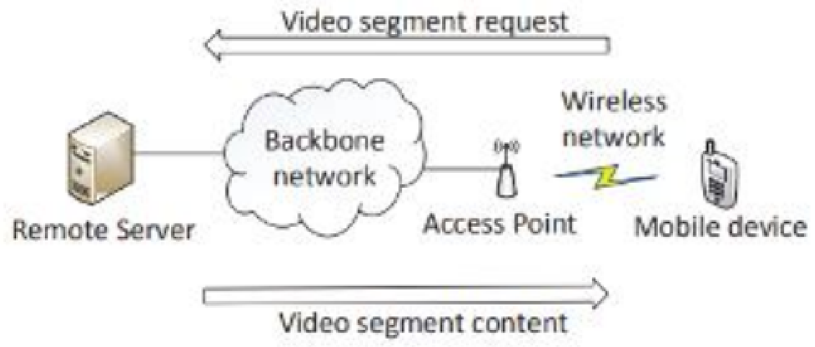


Figure 2. Structure of adaptive bitrate streaming over wireless networks [12]

Dynamic Adaptive Streaming over HTTP (DASH) or MPEG-DASH, which employs ABR streaming, has emerged as the predominant standard for video traffic on the Internet, as per recent statistics [13]. The DASH streaming standard has enabled more efficient video streaming, contingent upon the quality of the connection established between the server and client. DASH-encoded videos are segmented into smaller units, which can be represented in diverse formats. To summarise, the DASH protocol partitions videos into segments, where each segment represents a consistent proportion of the complete video's duration. Furthermore, it is possible to provide various video quality alternatives for every individual video segment [7].

2. Related Works

This section provides a review of existing adaptive multimedia streaming models that are appropriate for mobile cloud computing. In Ref. [14], an architecture for mobile video streaming, namely AMES-Cloud, was introduced. The technique offers a resolution to the issue of substandard performance in video transmission over wireless networks, including sporadic interruptions and extended buffering periods. The AMES-Cloud system comprises two distinct components, namely adaptive mobile video streaming (AMoV) and efficient social video sharing (ESoV). The utilisation of these two components serves to enhance the QoS for mobile video streaming. Adaptive streaming optimises the user experience by dynamically adjusting the streaming bit rate in response to changes in link capacity. Furthermore, a flexible and dynamic scheduling methodology for cloud-based video transcoding utilising MPEG-DASH was introduced by Legrand et al. [15]. The scheduler continuously monitors the utilisation of processors and assigns high-priority tasks to the fastest processors available in the cloud. Additionally, the selection of the video transcoding mode considers the system load. The experimental results indicate that the scheduler performs effectively concerning video completion time, system load balancing, and seamless playback of videos. In another investigation, scholars formulated a strategy to allocate the workload of media streaming solicitations across numerous servers based in the cloud. The implementation of horizontal elasticity involves the dynamic addition or removal of virtual media servers in response to demand and elasticity policies. Empirical evidence gathered from a genuine testbed indicates that the load balancer exhibits comparable efficacy to HAProxy in terms of various performance metrics such as jitter, packet loss rate, CPU utilisation, and RAM utilisation. Conversely, the elastic methodology exhibits robustness in managing variable workloads, rendering it well-suited for Content Delivery Networks (CDNs) in cloud computing [16]. Specifically, in Refs. [7, 17] a segment-aware adaptation (SARA) algorithm is proposed. The authors have observed that content with identical bitrates can exhibit varying chunk sizes. The Media Presentation and Description (MPD) file was updated to include information regarding the sizes of the chunks. The utilisation of weighted harmonic mean was employed to approximate throughput. The ABR adaptation logic algorithms were categorised into three groups by the researchers [18]. The adaptation logic that selects the bitrate of the next chunk is based on bandwidth estimations. The adaptation mechanism in question employs a buffer occupancy-based approach, whereby the modelling of the bitrate for the subsequent chunk is predicated on the buffer's level of occupancy. The adaptation logic employs hybrid techniques that involve the utilisation of both bandwidth estimation and buffer occupancy. Various adaptation logics and schedulers were evaluated on cellular LTE networks. The utilisation of proportional fairness as an LTE video streaming scheduler is deemed optimal.

A virtualized CDN was developed by the author in Ref. [19] with the support of Big Data technology and intended for implementation in telecom cloud deployments. The leaf cache nodes of the telecom CDN are situated close to ending users and are tasked with managing access to Video on Demand (VoD) content as well as adjusting live TV channels to suit the devices of individual users. Meanwhile, the intermediate cache nodes receive live TV channels and generate VoD content. MPEG-DASH is a technology that enables multiple HTTP servers to provide both live and stored media content to end-users from a virtualized leaf cache node. Through rigorous modelling within a realistic context, it has been determined that a slight reconfiguration of the CDN can result in a reduction of HTTP server allocation and bandwidth by 83.5%. This optimisation can be achieved without compromising the user experience, as evidenced by consistent video quality metrics. A strategy for rate adaptation in the context of fractional dynamic adaptive streaming over HTTP (FDASH) is presented in Ref. [20]. The system employs fuzzy logic to ascertain the optimal duration for buffering a video before its transmission to the client, the appropriate resolution for the video, the frequency of resolution variations, and the maximum duration for uninterrupted video playback. Research has demonstrated that the utilisation of FDASH can provide clients with the most favourable video rates, thereby preventing buffer overflow and unnecessary modifications to video resolution in response to changes in connection throughput. The Vsync method was introduced in REf. [21] as a technique for achieving synchronisation of mobile video content through cloud computing. The use of a cloud-based framework for real-time transcoding and transmission guarantees a superior level of video quality during video streaming. Vsync exhibits superior performance compared to other schemes by a substantial margin (80%) due to its utilisation of predictive models for video transcoding sessions and a QoS-based adaptive video streaming protocol. The study conducted in Ref. [22] evaluated the efficacy of a decentralised DASH video streaming approach. The authors devised a technique for selecting a server on the client side with the aim of optimising bandwidth utilisation and ensuring superior video playback quality. Consequently, a methodology was developed to select servers that considered both network latency and bandwidth. The findings indicate that the latency-based approach can effectively maintain optimal buffer levels even in the face of fluctuating bandwidth conditions originating from content servers. Tsilimantos *et al* aimed to passively estimate parameters of low-level HTTP Adaptive Streaming (HAS) applications through their trace profiling method. Through the monitoring of incoming IP packets, it was ascertained that the machine learning system can differentiate between various categories of video streams, thereby enabling it to precisely assess the buffering status of an HTTP adaptive streaming (HAS) client's playback in real-time. Furthermore, it was observed through experimentation with the YouTube mobile client that the aforementioned method exhibited high precision in the face of diverse link quality fluctuations [23]. The authors in Ref. [13] have presented a novel method named Requet, which is capable of real-time detection of the encrypted trace's quality. The utilisation of three QoS parameters, namely buffer warning, video state, and video quality, was relied upon to enable real-time allocation of resources at the network stage, owing to their significant role in this regard. To enhance the ABR trace data, a video state labelling system was devised to automatically generate accurate labels for ground truth. The experimental findings indicate that Requet's predictive capability was significantly enhanced when utilising chunk-based features in contrast to an IP-layer feature-based baseline system.

Numerous factors, such as sporadic disruptions, buffering issues, bandwidth fluctuations, and constrained bandwidth, continue to prevent the optimal distribution of media content through cloud computing. Despite research endeavours aimed at enhancing QoS and QoE in multimedia cloud computing, these challenges persist. This paper introduces an improved dynamic streaming system designed for cloud-based media delivery.

3. Methodology

The proposed methodology employs the DASH protocol, utilising cloud computing technology in conjunction with a player known as xenplayer, which serves as a decoder and facilitates the uninterrupted transmission of video content over mobile wireless networks. The proposed scheme is founded on DASH, which was selected due to its distinctive capability to address the constraints associated with conventional progressive downloads, including excessive bandwidth consumption in aggressive buffering and insufficient bandwidth diversity. Moreover, DASH exhibits seamless traversal of firewalls, NAT devices, and middle-boxes while demonstrating compatibility with diverse mobile devices. The DASH protocol entails encoding raw video into distinct representations, also known as bit rates, each of which is subsequently partitioned into segments of uniform duration. The adaptation algorithms utilised by Xenplayer are designed to dynamically modify the playback bit rate based on the measured bandwidth to optimise the playback quality for a given network bandwidth. The concept of adaptive video streaming involves adjusting the necessary bandwidth of a video stream to match the available throughput along the network path between the stream's source and the client. Adaptation is achieved through modulation of the bit rate of the streamed video, consequently impacting its visual quality.

The bit rate refers to the number of bits necessary to encode a single second of video. One strategy employed involves segmenting the video stream and subsequently encoding each segment at multiple quality levels. The designations used to denote the standards of excellence are commonly referred to as representations. Clients may request sequential chunks at multiple levels of quality to adapt to the fluctuating network conditions, based on the predicted functional throughput. The algorithm has been incorporated into the client. The process of algorithmic determination of segment representation contributes to the mitigation of buffering, thereby enhancing the quality of the viewing experience. Xenplayer is an Android-based media player that utilises the foundational Android API. This alternative media player supplants the conventional Android framework media player by accommodating DASH technology. Xenplayer utilises the media encoder and audio track classes within the Android framework through its pre-existing video renderers. The utilised video source originates from the animated short film entitled "Big Buck Bunny" and comprises five distinct video representations. The video has been uploaded to the Microsoft Azure cloud server. The characteristics of the video source are described in Table 1.

Table 1. Video Source File description

File Description	Unit
File Type	MP4
File Size	20MB
Audio Format	mp4a
Audio Channels	3
Audio Bits Per Sample	16
Audio Sample Rate	48000
Average Bitrate	1.43 Mbps
Image Size	1280 X 20
Mega Pixels	0.922
Media duration	0:01:57
Bit Depth	24
Video Frame Rate	25
X Resolution	72
Y Resolution	72

Consider a video stream comprising segments, where each segment has a playback duration of t seconds. Various representations (R) are accessible for each segment. The DASH protocol enables clients to obtain a set of available representations before streaming. This is achieved by downloading the Media Presentation Description (MPD) file based on the extended markup language (XML). Upon the user's request to access the video stream, the client proceeds to download the segments sequentially, with each segment being downloaded in a single representation and without any preemption during the download process. Upon downloading, the information undergoes decoding and is subsequently displayed to the end user. Complete downloading of a segment is not a prerequisite for its decoding and playback. The decoder typically requires a minimum quantity of bytes to initiate decoding, commonly lower than the segment's length. The present study centres on the dynamic selection of a representation from R, utilising the throughput-based adaptation algorithm, which involves estimating the necessary bandwidth for the given segment. The subsequent segment of this discourse deliberates on the system architecture as illustrated in Figure 3.

3.1. *DASH Cloud Server*

The DASH cloud server is equipped with video files. The video sequence is transferred to the cloud server and subsequently subjected to encoding. The process of encoding involves the partitioning of the video sequence into varying sizes of smaller segments. Following the encoding process, the generation of an MPD file takes place. The MPD file is an XML-based document that encompasses pertinent details regarding media segments, their interrelationships, and requisite information for selecting among them, along with other metadata that may be deemed essential by the client.

3.2. *DASH Client*

The DASH client is comprised of several components, including the adaptation logic, MPD parser, segment parser, player buffer, and media player. A cloud server is queried through an HTTP Get request to verify the presence of the video sequence within the server. If the video is hosted on a server, the transmission control protocol (TCP) is utilised to convey the media presentation description (MPD) file to the recipient.

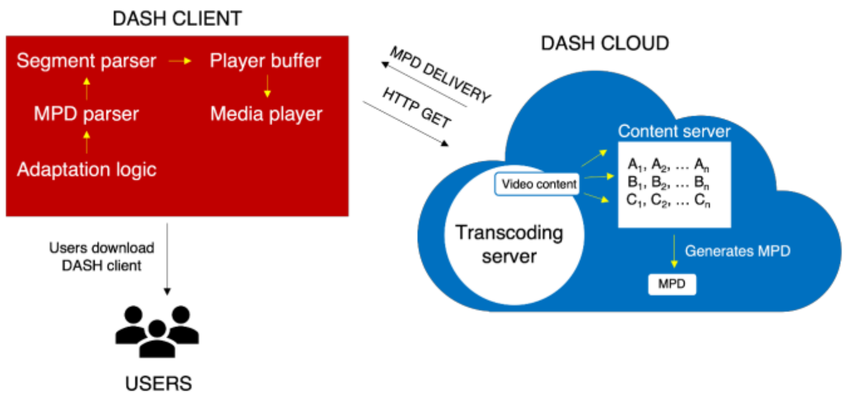


Figure 3. Proposed system architecture

The adaptation logic assesses the bandwidth that is currently accessible. The MPD parser verifies the conformity of the MPD file format to guarantee its accurate delivery. The MPD file is analysed by the segment parser to verify that the segment base and representation comply. The validity of the index is confirmed. If the recipient obtains information at a rate that surpasses the predetermined threshold, the surplus data is retained in a buffer. The multimedia data is processed by the media player and subsequently transformed into either visual or audio output, contingent upon the specific multimedia format.

3.3. *XENPLAYER Adaptation Scheme*

The adaptation scheme highlights the process by which the video sequence is procured from the cloud server and disseminated to the client in reduced segments that are contingent upon fluctuations in bandwidth. The process of inputting the video URL is delineated in Algorithm 1. If the video sequence is located on the server, the client proceeds to download the MPD file. The downloading of chunks is contingent upon the client's available bandwidth. The process of estimating the bandwidth is explained in Algorithm 2. The sliding percentile technique is employed to obtain an approximation of the subsequent bandwidth. A weight limit is established to ensure that the available portions do not surpass the predetermined weight threshold. The first-in, first-out (FIFO) scheduling technique is utilised to accommodate additional weight. The weights have been arranged in a sorted manner, and the 0.5 percentile has been derived. At time t, Algorithm 3 is executed promptly following the retrieval of segment a (t). This guarantees that the system effectively adjusts the video resolution to the varying levels of available bandwidth. The maximum initial bit rate (MIB) represents the upper limit of the bit rate that is recommended to be used in the absence of a bandwidth estimate. The Minimum Duration for Quality Increase (MDQI) refers to the minimum amount of buffered data necessary for the selected track to transition to a higher quality level. The MDQD, or maximum duration for quality decrease, refers to the maximum duration of buffered data that is necessary for a selected track to transition to a lower quality level.

```
     Algorithm 1: XENPLAYER
1.   START
2.   Enter the video URL
3.   Client sends request to server
4.   Repeat 2 until == true
5.   Server sends manifest file
6.   Client reads manifest file
7.   Client checks device bandwidth
8.   If device bandwidth == 2G device —— bandwidth == 3G device
     Then activates 2G segment —— 3G segment
     else goto 10
9.   Repeat 7 until videolength = -1
10.  STOP
```

```
Algorithm 2: BANDWIDTH ESTIMATION
Input:
streamCount, bytesReceived, startTime, end Time
Procedure:
1. START
2. SET K,L= Ψ
3. SET streamcount, bytesreceived = 0
4. If chunk transfer state == begin, goto 5, else 19
5. SET start Time = SystemClock.elapsedRealtime()
6. streamcount = streamcount + 1
7. bytesreceived = bytesreceived + 1
8. If chunk transfer state == active, go to 6 else 9
9. SET endTime = SystemClock.elapsedRealtime()
10. SET elaspedTime = endTime - start Time
11. total Time = total Time + elapsed Time
12. totalBytesReceived = totalBytesReceived + bytesreceived
13. If elasped Time > 0, go to 14 else 19
14. Convert to bitspersecond (bitsPerSecond = (bytesreceived * 8000) / elapsed Time))
15. Slidingpercentile.addSample( $\sqrt{\text{bytes received}}$), bitsPerSecond)
16. If total Time ≥ K || totalBytesReceived ≥ L, go to 17 else 19
17. SET BitrateEstimateFloat = slidingpercentile.getpercentile(0.5)
18. If BitrateEstimate is NaN, set BitrateEstimate to NoEstimate else, BitrateEstimate = BitrateEstimateFloat,
19. STOP
End Procedure:
Output:
BitrateEstimate, NoEstimate
```

4. Result and Evaluation

This section presents the outcomes of the Java programming language implementation and experimentation within cloud computing technology. Furthermore, the evaluation results that rely on particular performance metrics are also

```
Algorithm 3 BANDWIDTH SELECTION
Input:
$\Psi_i = a(t)$ (set of past available representations)
Procedure:
1. START
2. SET mdqi = $\beta$
3. SET mdqd = $\alpha$
4. SET b d uration = $\gamma$
5. while ($s_b$itrate ! = null)
6. if sbitrate, cbitrate and bduration < mdqi then sprocess ¸ false
7. else if sbitrate, cbitrate and bduration ≥ mdqd then sprocess ¸ false
8. else if sbitrate > cbitrate and bduration > mdqi then sprocess ¸ true
9. else if sbitrate < cbitrate and bduration ≤ mdqd then sprocess ¸ true
10. END IF
11. END WHILE
12. STOP
End Procedure:
Output:
$\Psi_i = a(t)$ (set of past available representations)
```

presented.

4.1. Delivering Video-On-Demand With Azure Media Service

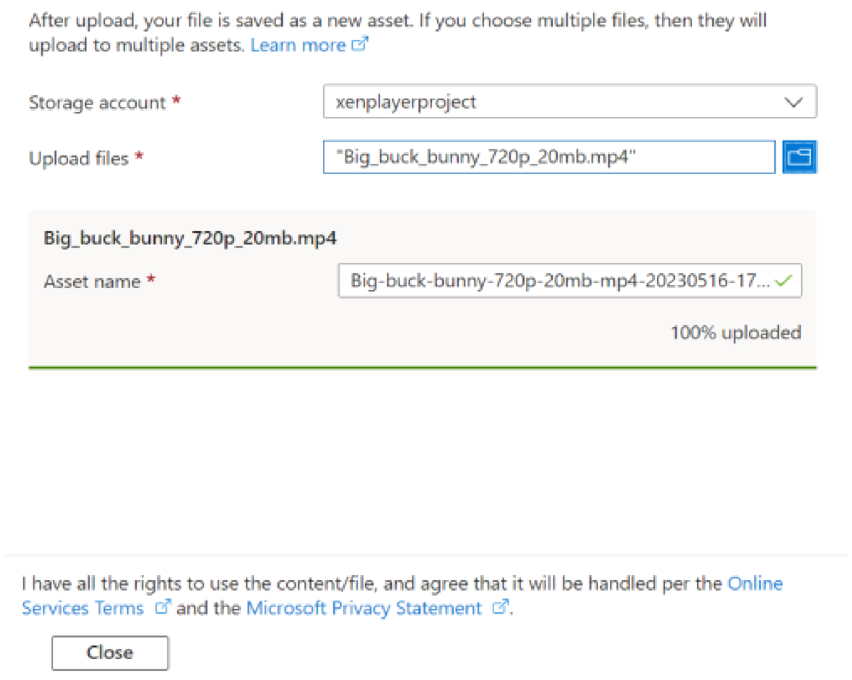
a) **Streaming endpoint**: To dynamically package the video content, the streaming endpoint must be in a running state of operation before uploading the video to the Azure media service. The process of initiating the streaming end-point entails the following steps:
   i) Sign in to the Azure portal.
   ii) Select Settings, and go to streaming endpoints.
   iii) Select the default (endpoint. item), then, select the start icon.
   iv) Select the save button.

b) **Upload and view the video**: Before encoding the video into multiple bitrates, it is necessary to upload the video file to Azure Media Services, as depicted in Figure 4. The video files will be transferred to the designated repository for storage and management. The process of uploading files involves the following procedures:
   i) In the Azure portal, select Azure Media Service account.
   ii) Select settings, go to the asset, and select the upload button

   Figure 5 illustrates a representation of the characteristics of the video asset. The process of accessing the attributes of the uploaded video asset involves the following steps:
   i) In the Azure portal, select Azure Media Service account.
   ii) Select settings and select the asset. The new asset is listed in the asset pane
   iii) Select the name of the uploaded video asset.

c) **Video asset encoding:** Monitoring the encoding process can be achieved through the selection of appropriate settings and jobs. Figure 6 presents an illustration of the encoding process. The process of encoding video content using Media Encoder Standard within the Azure portal necessitates the following procedure:
   i) In the Azure portal, select the Azure Media Services account.
   ii) Select Settings and select the asset to encode.
   iii) Select the Encode button.
   iv) In the Encode asset pane, select the Media Encoder
   v) Standard processor and the Content Adaptive Multiple Bitrate MP4 pre-set.

d) **View and publish encoded video:** Figure 7 depicts the encoded video fragments. The following procedure is required to view the encoded video segments:



After upload, your file is saved as a new asset. If you choose multiple files, then they will upload to multiple assets. Learn more

Storage account *	xenplayerproject
Upload files *	"Big_buck_bunny_720p_20mb.mp4"

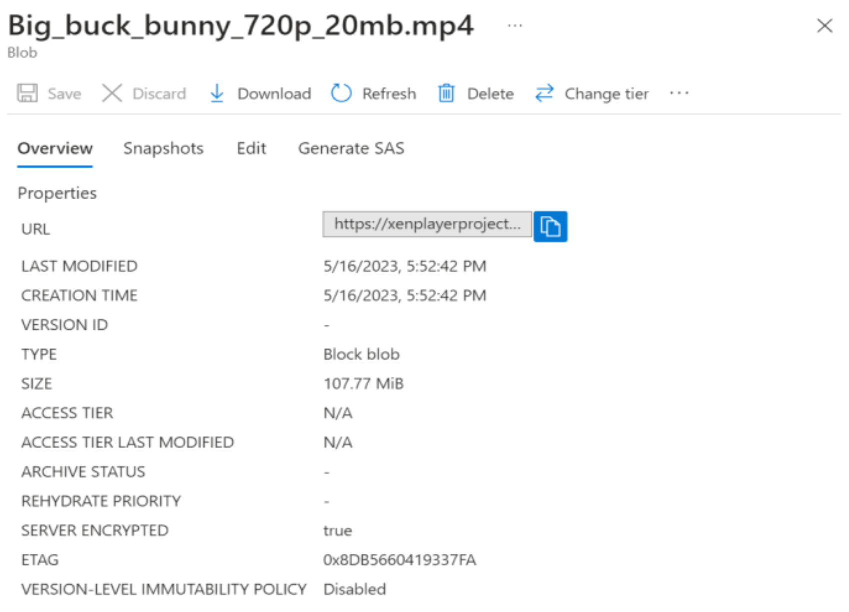
Big_buck_bunny_720p_20mb.mp4

Asset name *	Big-buck-bunny-720p-20mb-mp4-20230516-17...

100% uploaded

I have all the rights to use the content/file, and agree that it will be handled per the Online Services Terms and the Microsoft Privacy Statement.

Figure 4. Uploading video file to the asset folder.



Big_buck_bunny_720p_20mb.mp4

Blob

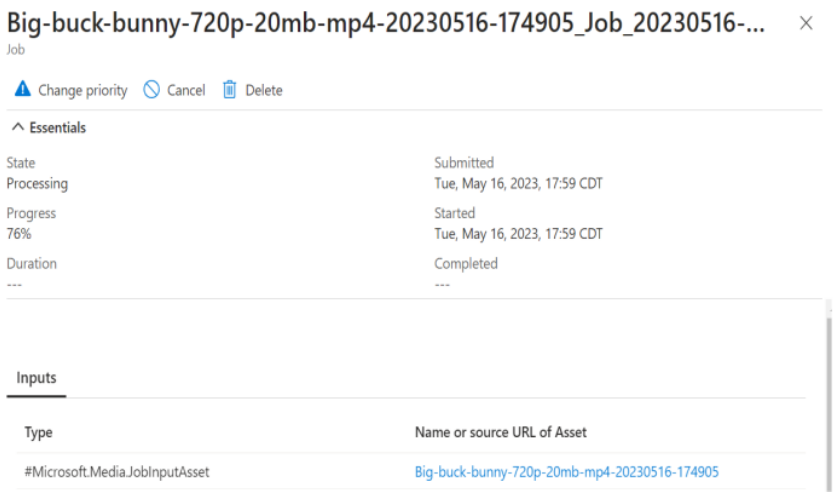
Properties

URL	https://xenplayerproject...
LAST MODIFIED	5/16/2023, 5:52:42 PM
CREATION TIME	5/16/2023, 5:52:42 PM
VERSION ID	-
TYPE	Block blob
SIZE	107.77 MiB
ACCESS TIER	N/A
ACCESS TIER LAST MODIFIED	N/A
ARCHIVE STATUS	-
REHYDRATE PRIORITY	-
SERVER ENCRYPTED	true
ETAG	0x8DB5660419337FA
VERSION-LEVEL IMMUTABILITY POLICY	Disabled

Figure 5. Properties of the video asset.

i) In the Azure portal, select the Azure Media Services Account
ii) Select Settings, and go to Assets,
iii) Select the asset that you want to view encoded segments.

Figure 8 presents a selection of encoded video that has been published. To furnish the user with a uniform resource


Big-buck-bunny-720p-20mb-mp4-20230516-174905_Job_20230516-...

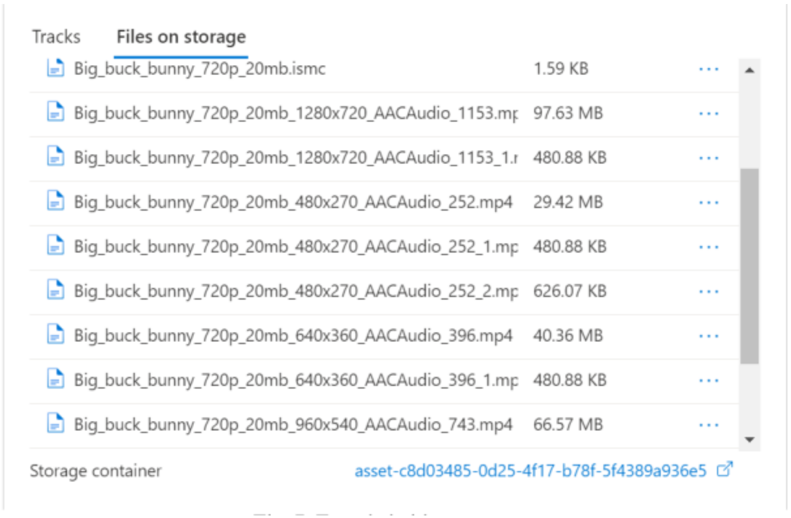
Job

State: Processing
Submitted: Tue, May 16, 2023, 17:59 CDT
Progress: 76%
Started: Tue, May 16, 2023, 17:59 CDT
Duration: ---
Completed: ---

Type	Name or source URL of Asset
#Microsoft.Media.JobInputAsset	Big-buck-bunny-720p-20mb-mp4-20230516-174905

Figure 6. Encoding the video asset

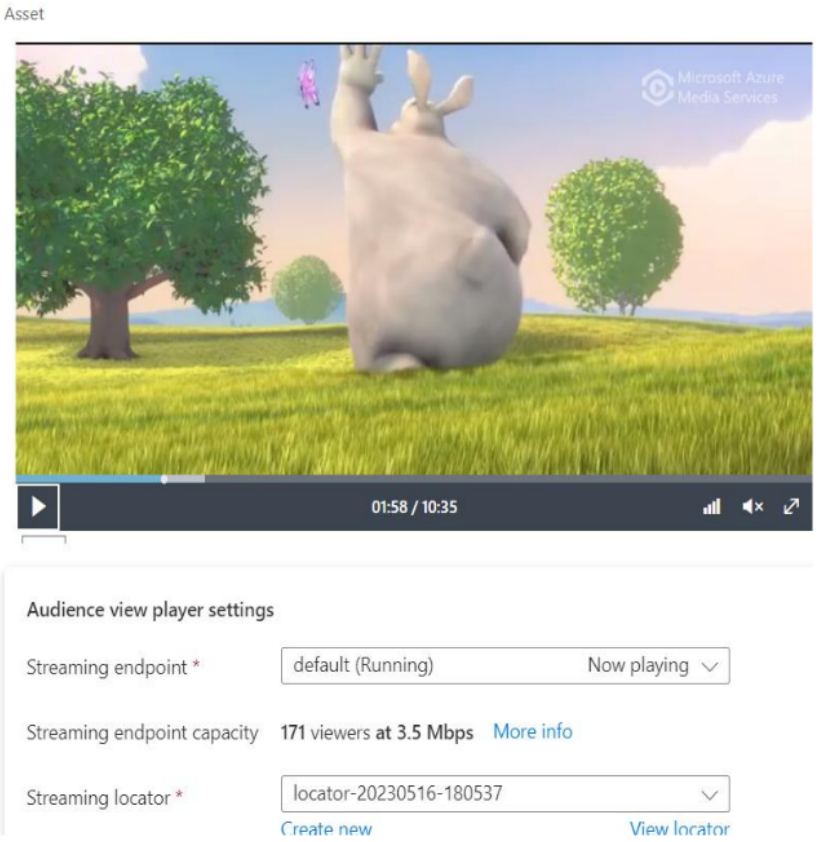
locator (URL) that is capable of facilitating the streaming of video content, a locator is initially generated. A locator facilitates the retrieval of files that are contained within an asset. Azure Media Services facilitates two distinct categories of locators, namely streaming and progressive locators. Streaming locators are used in the context of adaptive streaming. Instances of adaptive streaming encompass HTTP Live Streaming, Smooth Streaming, and MPEG-DASH. Progressive locators, on the other hand, are employed for dispensing video sequences through progressive download. To construct a URL for MPEG-DASH streaming, one must append the format ”mpd-time-csf” to the URL. The textual representation denoting the current state of the video, indicating whether it is in the ready or buffering state, is referred to as the status text. The analytics view presents data about each downloaded chunk, including but not limited to the URL, start time, end time, bit rate, and delay. The aforementioned also indicates the duration of the start-up latency that transpired before the successful loading of the video content onto the media player.



Big_buck_bunny_720p_20mb.ismc	1.59 KB
Big_buck_bunny_720p_20mb_1280x720_AACAudio_1153.mp	97.63 MB
Big_buck_bunny_720p_20mb_1280x720_AACAudio_1153_1.r	480.88 KB
Big_buck_bunny_720p_20mb_480x270_AACAudio_252.mp4	29.42 MB
Big_buck_bunny_720p_20mb_480x270_AACAudio_252_1.mp	480.88 KB
Big_buck_bunny_720p_20mb_480x270_AACAudio_252_2.mp	626.07 KB
Big_buck_bunny_720p_20mb_640x360_AACAudio_396.mp4	40.36 MB
Big_buck_bunny_720p_20mb_640x360_AACAudio_396_1.mp	480.88 KB
Big_buck_bunny_720p_20mb_960x540_AACAudio_743.mp4	66.57 MB

Storage container: asset-c8d03485-0d25-4f17-b78f-5f4389a936e5

Figure 7. Encoded video segments



Audience view player settings

Streaming endpoint * default (Running) Now playing

Streaming endpoint capacity 171 viewers at 3.5 Mbps More info

Streaming locator * locator-20230516-180537

Figure 8. Publish encoded video.

5. Performance Testing

The system's performance was assessed and examined by implementing network scenarios characteristic of low-speed network environments, with Nigeria serving as a representative case study. The study conducted measurements on the startup delay, end-to-end delay, and bytes downloaded in both 2G and 3G networks. A visual depiction of the results was procured and documented.

5.1. *Second Generation Network (2G)*

The 2nd generation (2G) cellular network provides a peak data transfer rate of up to 50 kilobits per second (Kbps). Additional 2G technologies that have been implemented comprise General Packet Radio Service (GPRS), which provides a transfer rate of 30 Kbps to 40 Kbps, and GSM Revolution (EDGE), which offers enhanced data and a transfer rate of 100 Kbps to 120 Kbps. The results of the evaluation indicate that the start-up delay achieved through the implementation of 2G amounts to 13 seconds. Upon integration into a 2G network, the video sequence required 13 seconds to load appropriately owing to the restricted bandwidth. The temporal duration from the initiation to the completion of transmission for the initial six data segments exhibited periodic fluctuations, ranging from 8 seconds to 10 seconds, as depicted in Figure 9. The minimum duration of end-to-end delay was observed during the sixth chunk, with a delay of 6 seconds. Figure 10 displays that the minimum byte downloaded is 58.42 kbps. The result indicates that a shorter delay leads to a reduced number of downloaded bytes.

5.2. *Third Generation Network (3G)*

The 3rd Generation (3G) cellular network provides a peak data transfer rate of up to 21.6 megabits per second (Mbps). Several 3G technologies have been implemented, such as Universal Mobile Telecommunication System

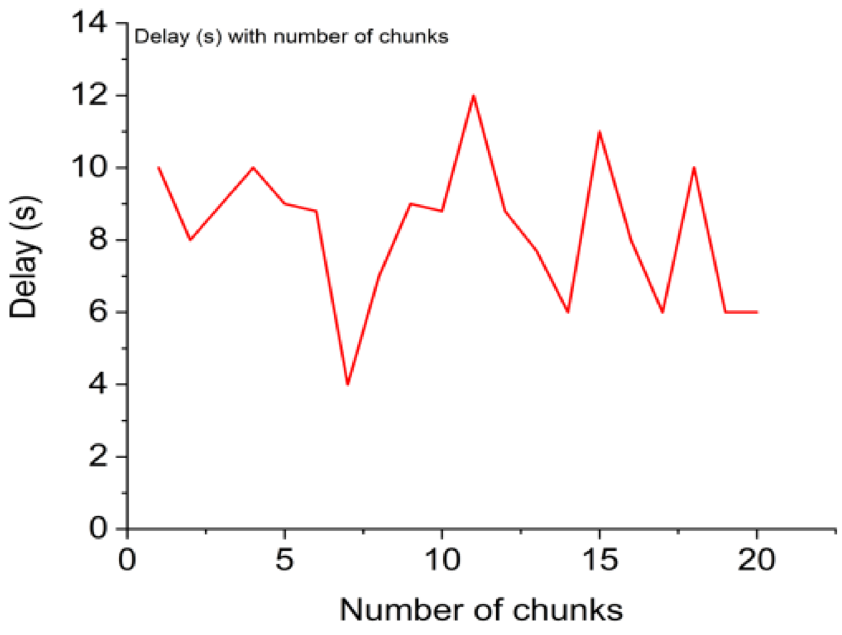


Figure 9. End-to-end delay in 2G network

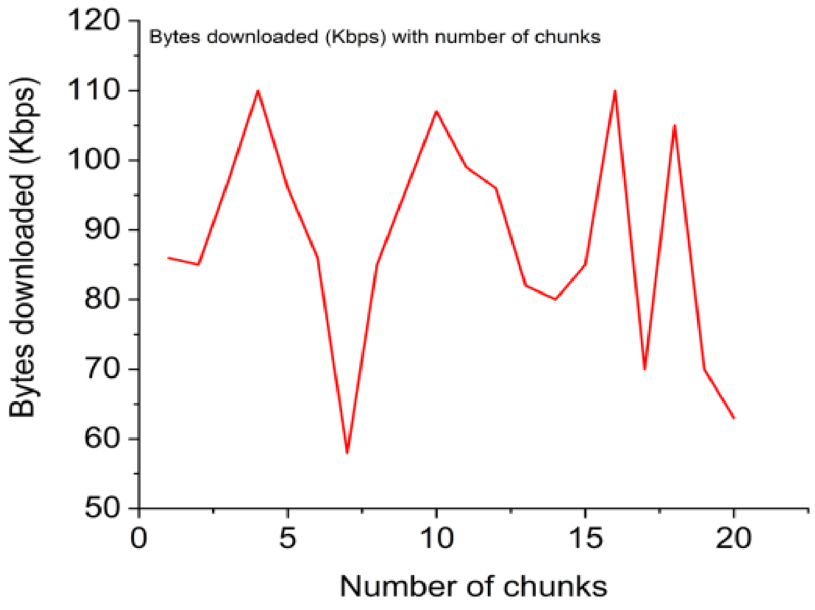


Figure 10. Bytes downloaded in 2G network

(UMTS), boasting a transfer rate of 2 Mbps, High-Speed Upload Packet Access (HSPA+), with a transfer rate of 21.6 Mbps, and Wide-band CDMA (WCDMA), providing a transfer rate of 384 kbps. A delay of 3 seconds was observed in the initiation of the 3G technology implementation. The video sequence was observed to load appropriately after a duration of 3 seconds on a 3G network, as depicted in Figure 11. A delay of one second was observed in the stabilisation of the initial four chunks. The fifth segment exhibits a temporal lag of zero seconds, which is sustained for the subsequent three segments. Figure 12 illustrates that the sixth chunk experienced no delay and resulted in the minimum number of bytes downloaded. The simulation results indicate that the proposed scheme effectively managed bandwidth variation and successfully handled pre-recorded video, resulting in uninterrupted and seamless streaming.

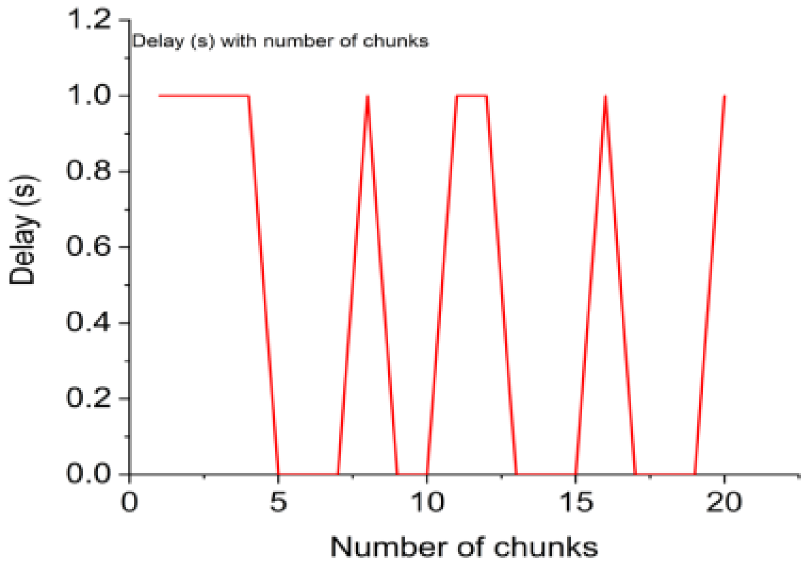


Figure 11. End-to-end delay in 3G network

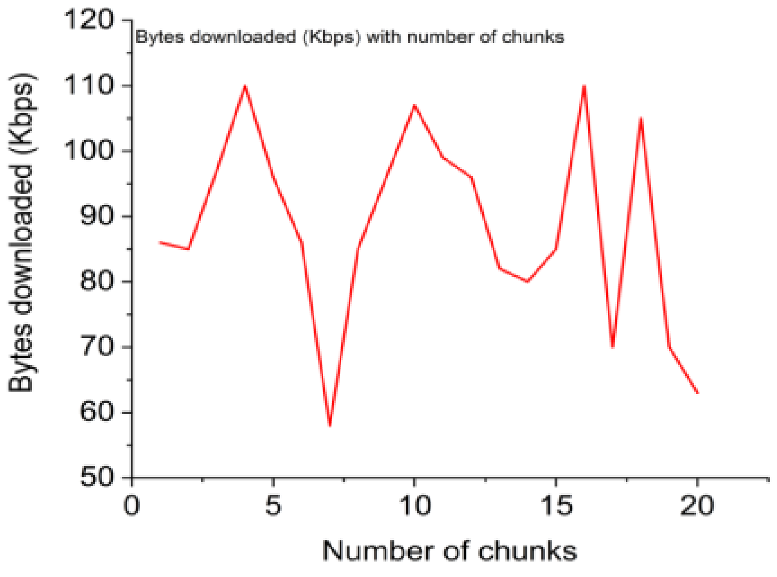


Figure 12. Bytes downloaded in 3G network

6. Conclusion

The present study introduces an effective adaptive streaming methodology designed for mobile cloud applications. The aforementioned methodology was executed by utilising the Java programming language in conjunction with Microsoft Azure, which served as the cloud computing platform. The proposed scheme exhibits efficient performance regardless of fluctuations in bandwidth. Furthermore, the aforementioned scheme attained a notable streaming rate and sustained playback continuity. The system demonstrated the capability to effectively manage pre-recorded video content across diverse network environments, ensuring seamless streaming devoid of any interruptions or disturbances. The system successfully rendered high-quality video content from the user's point of view. It is imperative to note that the suggested streaming methodology effectively tackled certain challenges hindering the implementation of video applications in mobile cloud computing under low-speed conditions, particularly prevalent in numerous developing nations. Nevertheless, the technique can be improved further to reduce the start-up and end-to-end delays in 2G networks. Moreover, additional investigation could be carried out to evaluate the efficacy of this approach in more consistent network settings, such as 4G and 5G networks.

References

[1] X. Wang, M. Chen, T.T. Kwon, L. Yang & V.C. Lee, "Ames-cloud: A framework of adaptive mobile video streaming and efficient social video sharing in the clouds", IEEE Transactions on Multimedia **15** (2013) 811. https://doi.org/10.1109/TMM.2013.2239630.

[2] B. Wickremasinghe, R.N. Calheiros & R. Buyya, "Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications", Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference (2010) 446. `https://doi.org/10.1109/AINA.2010.32`.

[3] A. Weiss, "Computing in the clouds", Networker **11** (2007) 16. http://dx.doi.org/10.1145/1327512.1327513.

[4] D. Tsilimantos, T. Karagkioules & S. Valentin, "Classifying flows and buffer state for youtube's http adaptive streaming service in mobile net-works", Proceedings of the 9th ACM Multi-media Systems Conference (2018) 138. `https://doi.org/10.1145/3204949.3204955`.

[5] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica & M. Zaharia, "A view of cloud computing", Communications of the ACM **53** (2010) 50. http://dx.doi.org/10.1145/1721654.1721672.

[6] C.S. Yeo & R. Buyya, "Integrated risk analysis for a commercial computing service in utility computing", Journal of Grid Computing **7** (2009) 1. https://doi.org/10.1007/s10723-008-9103-2.

[7] D.K. Krishnappa, D. Bhat & M. Zink, "Dashing youtube: An analysis of using dash in youtube video service" 38th Annual IEEE Conference on Local Computer (2013) 407. https://doi.org/10.1109/LCN.2013.6761273.

[8] H. Ma, B. Seo & R. Zimmermann, "Dynamic scheduling on video transcoding for mpeg dash in the cloud environment" Proceedings of the 5th ACM Multimedia Systems Conference **283** (2014) 283. https://doi.org/10.1145/2557642.2557656.

[9] B. Al-Madani, A. Al-Roubaiey & Z.A. Baig, "Real-time qos-aware video streaming: a comparative and experimental study", Advances in Multimedia 2014, 1 (2014). https://doi.org/10.1155/2014/164940.

[10] G.J. Conklin, G.S. Greenbaum, K.O. Lillevold, A.F. Lippman & Y.A. Reznik, "Video coding for streaming media delivery on the internet", IEEE Transactions on Circuits and Systems for Video Technology **11** (2001) 269. https://doi.org/10.1109/76.911155.

[11] M. Ruiz, M. German, L.M. Contreras & L. Velasco, "Big data-backed video distribution in the telecom cloud", Computer Communications **84** (2016) 1. http://dx.doi.org/10.1016/j.comcom.2016.03.026.

[12] K. Dong, J. He & W. Song, "Qoe-aware adaptive bitrate video streaming over mobile networks with caching proxy", 2015 International Conference on Computing, Networking and Communications (ICNC) (2015) 737. https://doi.org/10.1109/ICCNC.2015.7069438.

[13] C. Gutterman, K. Guo, S. Arora, X. Wang, L. Wu, E. Katz-Bassett & G. Zussman, "Requet: Real-time qoe detection for encrypted youtube traffic", Proceedings of the 10th ACM Multimedia Systems Conference (2019) 48. http://dx.doi.org/10.1145/3304109.3306226.

[14] F. Wamser, S. Hofner, M. Seufert & P. Tran-Gia, "Server and content selection for mpeg dash video streaming with client information", Proceedings of the Workshop on QoE-based Analysis and Management of Data Communication Networks (2017) 19. https://doi.org/10.1145/3098603.3098607.

[15] A. Legrand, L. Marchal & H. Casanova, "Scheduling distributed applications: the sim-grid simulation framework", CCGrid 2003, 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (2003) 138. https://doi.org/10.1109/CCGRID.2003.1199362.

[16] J.J. Quinlan, A.H. Zahran & C.J. Sreenan, "Datasets for avc (h. 264) and hevc (h. 265) evaluation of dynamic adaptive streaming over http (dash)", Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16), Association for Computing Machinery (2016) 51. https://doi.org/10.1145/2910017.2910625.

[17] P. Juluri, V. Tamarapalli & D. Medhi, "Sara: Segment aware rate adaptation algorithm for dynamic adaptive streaming over http", 2015 IEEE International Conference on Communication Workshop (ICCW) (2015) 1765. https://doi.org/10.1109/ICCW.2015.7247436.

[18] A. Masood, M. Sharif, M. Yasmin & M. Raza, "Virtualization tools and techniques: Survey", Nepal Journal of Science and Technology **15** (2015) 141. https://doi.org/10.3126/njst.v15i2.12131.

[19] P.A. Rego, M.S. Bonfim, M.D. Ortiz, J.M. Bezerra, D.R. Campelo & J.N. de Souza, "An openflow-based elastic solution for cloud-cdn video streaming service", 2015 IEEE Global Communications Conference (GLOBE-COM) 1 (2015). https://doi.org/10.1109/GLOCOM.2015.7417789.

[20] L.M. Vaquero, L. Rodero-Merino, J. Caceres & M. Lindner, "A break in the clouds: towards a cloud definition", ACM SIGCOMM Computer Communication Review **39** (2008) 50. https://doi.org/10.1145/1496091.1496100.

[21] E. Baik, A. Pande, Z. Zheng & P. Mohapatra, "Vsync: Cloud based video streaming service for mobile devices", IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications (2016) 1. https://doi.org/10.1109/INFOCOM.2016.7524567

[22] D.J. Vergados, A. Michalas, A. Sgora, D.D. Ver-gados & P. Chatzimisios, "Fdash: A fuzzy-based mpeg/dash adaptation algorithm", IEEE Systems Journal **10** (2015) 859. https://doi.org/10.1109/JSYST.2015.2478879.

[23] Y. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu & B. Sinopoli, "Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction", Proceedings of the 2016 ACM SIGCOMM Conference (2016) 272. https://doi.org/10.1145/2934872.2934898.