Adding a new dimension to HTTP Adaptive Streaming through multiple-source capabilities

Joachim Bruneau-Queyreix, Mathias Lacaud, Daniel Negru, Jordi Mongay Battala, Eugen Borcoci

Adding a new dimension to HTTP Adaptive Streaming through multiple-source capabilities

Joachim Bruneau-Queyreix[1,2], Mathias Lacaud[2], Daniel Negru[1], Jordi Mongay Battala[3,4], Eugen Borcoci[5]

[1]University of Bordeaux, LaBRI, UMR 5800 F-33400 Talence, France
[2]Viotech Communications, Versailles, France
[3]National Institute of Telecommunications, Warsaw, Poland
[4]Warsaw University of Technology, Warsaw, Poland
[5]University Politechnica of Bucarest, Bucarest, Romania

Abstract

Adaptive streaming protocols over HTTP (HAS), such as MPEG-DASH, have become the de-facto solutions to deliver video over the Internet. By avoiding buffer stalling, HAS increases end-user's Quality of Experience (QoE). These solutions always bind a client to one server. We propose to extend HAS capabilities to multiple-source streaming, which offers the opportunity to obtain higher QoE by exploiting expanded bandwidth and link diversity in distributed streaming infrastructures. We expose a pragmatic evolving DASH-compliant solution for dynamic adaptive Multiple-Source Streaming over HTTP (MS-Stream) that simultaneously utilizes several servers. We validated our approach using network profiles from DASH Industry Forum and Internet users. Our solution is compared to optimal -yet practically unfeasible- DASH solutions in multiple-source environments and over several QoE criteria. Results show the QoE gains of using MS-Stream against DASH; an online demonstration of our work is available.

Keywords: DASH; adaptive streaming; multi-source streaming; evolving streaming solution.

1 INTRODUCTION

The demand on network resources is growing every day driven by the needs of end-users to consume more content with better quality over the Internet. Most of the time, this increase is not followed by the necessary upgrade of core networks capacity due to the important costs it incurs. A lot of technologies and architectures propose alternatives to overcome these limitations by providing a single intermediate with high throughput capacity and geographically close to end-users such as Content Delivery Networks (CDNs). Most of the single intermediate point solutions are using HTTP Adaptive Streaming (HAS) techniques where one uses a single server and adapts the requested content bitrate according to the estimated network throughput. Compared to non-adaptive streaming protocols, the adaptability and flexibility in HAS enable higher end-users' Quality of Experience (QoE) by avoiding video freezing events.

Such CDN-like solutions, based on the single-server approach, scale vertically and horizontally at expensive deployment and maintenance costs that eventually make their services prohibitively pricey for Content Providers (CPs) looking for high QoE content delivery. In order to alleviate this cost issue, we propose to stop confining

streaming to the single-server approach and propose an evolving streaming solution taking advantage of multiple types of content sources simultaneously, including cheap commodity devices (Set-Top boxes, end-users' devices, ISP's home-gateways) generally located at end-users' premises, with many of the costs covered by the end users). Indeed, hybrid systems (such as P2P-CDN) scale horizontally at cheaper costs compared to pure CDN solutions [1][2]. However, HAS in distributed environment has not yet been enabled in pragmatic-enough ways that would convince CPs to use alternative solutions to CDNs [3]. This paper is dedicated to the presentation of a pragmatic HAS solution that takes advantage of multiple-source environments achieving similar, generally higher, QoE compared to existing protocol: Multiple-Source Streaming over HTTP (MS-Stream). Figure 1 depicts the overall idea of MS-Stream compared to existing single intermediate point streaming.

MS-Stream is an evolution of HAS solutions (and more specifically, the Dynamic Adaptive Streaming over HTTP – DASH- standard) that simultaneously uses several servers for the download of one video segment. The resource provider periodically advertises clients with potential usable servers. MS-Stream clients request several servers to deliver sub-streams (referred as *descriptions*) generated from the existing set of DASH content representations so as to handle network-path heterogeneity. Descriptions creation represents very low CPU footprint operations. When retrieved, the requested descriptions are merged in order to reconstruct and display the original requested content quality. In the event of description loss or out-dated delivery, content playback continuity is not affected, only content quality is. Additionally, if the considered servers or paths suffer from degradation, content- and server-adaptations avoid QoE degradation. Thanks to its codec agnosticism and DASH-compliance, this proposal represents an evolving solution that can be applied to many scenarios (P2P, CDNs, Clouds, Set-Top-Box overlays as well as collaborations of resource providers to achieve higher QoE and create new businesses).

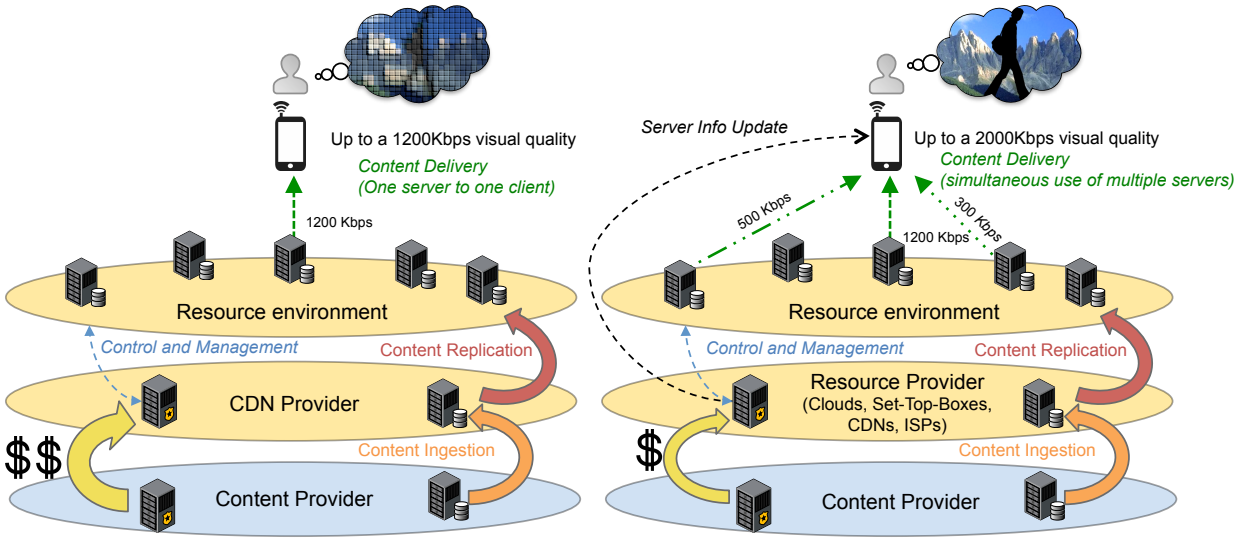

Figure 1: Comparison between single intermediate point streaming solutions (on the left) and our MS-Stream solution (on the right)

The contributions of the paper are described in section 2: the MS-Stream protocol architecture and messaging, the standard-compliant sub-stream generation scheme, and the MS-Stream's resiliency, flexibility and adaptability to resource heterogeneity. Section 3 exposes the MS-Stream's overheads and QoE gains in controlled networks and over the Internet. In section 4, we confront our work to other multi-source/path content delivery solutions. We conclude in section 5. The MS-Stream protocol has been implemented on top of the open-source dash.js client; an online demonstration is available [4].

2 MULTIPLE-SOURCE STREAMING OVER HTTP FOR ENHANCED QoE

We propose the Multiple-Source Streaming over HTTP protocol (MS-Stream) as a practical solution simultaneously exploiting multiple servers for consumers' QoE enhancement.

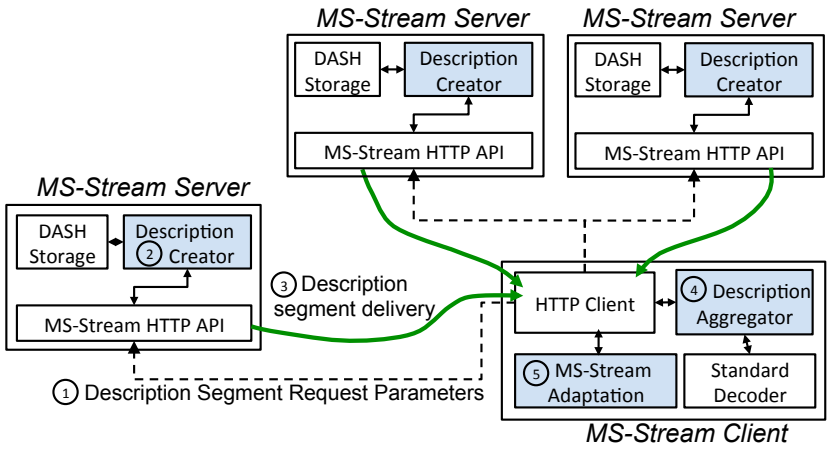

Figure 2: MS-Stream client/server architecture

2.1 System architecture and messaging

The MS-Stream overall client/server architecture is depicted in Figure 2 (additional modules -compared to DASH- are highlighted in blue). Prior the streaming session, a manifest file containing information about the available MS-Stream servers and the number of Group of Pictures per video segment (c.f.section 2.2) is delivered to the client. The MS-Stream content delivery steps include: (1) the client instructs MS-Stream servers to generate and deliver sub-streams through the MS-Stream HTTP API; (2) the *Description creator* generates the requested sub-stream from the existing set of content bitrate representations available in the *DASH Storage;* (3) descriptions transit on the network; (4) the *Description Aggregator* module merges the received descriptions so as to reconstruct the original content quality; Finally, (5) as content is being delivered over N paths (N sources), a global and per-path adaptation process is required to deal with path heterogeneity, performed in the *Adaptation* module. Hence, the MS-Stream client is an evolving DASH client, which incorporates a cost-effective description aggregation module and an adaptation engine capable of content and server adaptation.

From a technical standpoint, MS-Stream ensures DASH-backward compliance: upon the delivery of a regular DASH manifest file, an MS-Stream client can use uni-source DASH protocol. MS-Stream could be DASH-forward compliant quite easily: A DASH player would only have to accept our extended manifest file and could use MS-Stream servers as simple HTTP ones.

2.2 Sub-stream generation scheme

In order to benefit from codec standard-compliance and from a large amount of sub-streams (also referred as sub-segments or descriptions, c.f. related works), we focus on a hybrid sub-stream generation solution based on temporal and compressed data domains[5].

A sub-stream is generated by interleaving the Group Of Pictures (GoPs) available at two different bitrates of the same segment (Figure 3.a). The number of descriptions for a given video segment is an evolving parameter during the streaming session and is determined by the client according to the observed heterogeneous characteristics of the available servers, and to the targeted bitrate. Examples of GoP-based descriptions are depicted in Figure 3.c. Reconstructing the original content quality is achieved by selecting the GoPs of higher size in the pool of available descriptions (Figure 3b). Should some descriptions be missing for content reconstruction at client side, the content is still playable with sub-optimal visual quality, due to loss of information at the GoP level, inducing lower quality on a

GoP-duration basis).

We present a list of specificities included in our sub-stream generation scheme, easing its adoption by streaming providers: video-codec standard compatibility, tunable redundancy, low additional complexity and the possibility to create as many different descriptions as needed. By leaving unaltered the encoded video data and its data structure, descriptions are compliant with video codec standards and do not require new decoder implementations. Low-complexity post-encoding and pre-decoding steps are required to respectively create and merge descriptions, thus preserving standard compatibility. Descriptions are independent from each other; this is made possible by copying some common GoPs at a critically low bitrate (i.e., redundancy) into them. Such non-dependency between flows provides high reliability in heterogeneous unreliable networks, as any independent sub-segment can be lost without interrupting the streaming session. The more the redundancy is, the greater the network bandwidth consumption overhead. Consequently, in order to match any specific scenario (e.g., from reliable sources with high throughput to volatile sources with low throughput), MS-stream can embed mechanisms to control the degree of redundancy based on the observed network conditions and on the quality requested by the client.

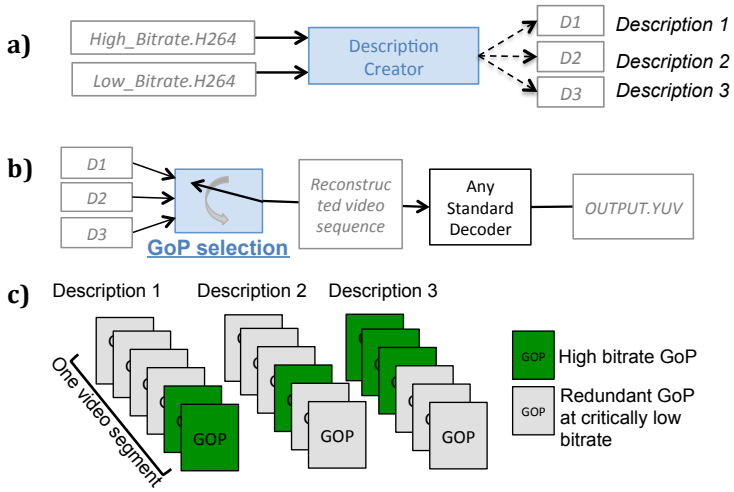

Figure 3: Multiple-Description scheme. a) Description generation, b) Description aggregation, c) example of GoP-based description sub-segments/descriptions

2.3 Multiple-Source Streaming protocol over HTTP

Multiple-Source Streaming over HTTP is an evolutionary protocol where a pragmatic usage of network throughput over multiple paths is achieve by the client: (1) simultaneous content retrieval from several servers, (2) GoP distribution to adapt descriptions to network heterogeneity, (3) content adaptation to increase QoE and reduce bandwidth overhead and (4) server adaptation to provide a flexible usage of network resources. Figure 4 presents the MS-Stream client's decision chart to specify sub-segments requests.

Simultaneous sub-segment retrieval from multiple sources and resiliency to resource heterogeneity

By simultaneously retrieving descriptions segments from several servers, the probability to receive at least one stream is increased. However, description synchronization is required so as to be resilient to network heterogeneity and avoid blocking events. A set of three rules has been designed for this synchronization at client side: (i) for a given content segment, if at least one description segment is retrieved, then other description downloads can be abandoned; this reactive rule ensures the delivery of at least one description segment before moving on to the next one; (ii) if the buffered content playout reaches a given lower threshold, description downloads can be canceled in order to ensure uninterrupted video experience, hence providing a temporary sub-optimal visual quality to the end-users; this second reactive rule can only be applied if rule (i) is satisfied; (iii) if the buffered content playout duration exceeds twice the average segment duration, then a timeout value is set on HTTP description segment requests. The timeout value reflects a consumption behavior (aggressive, conservative, etc.) and can be tuned during the streaming

session, according to the available buffered content. Once the timeout has elapsed, description requests can be canceled while satisfying rules (i) and (ii). This proactive rule enables the use of the buffer to compensate for network characteristic fluctuations on different path.

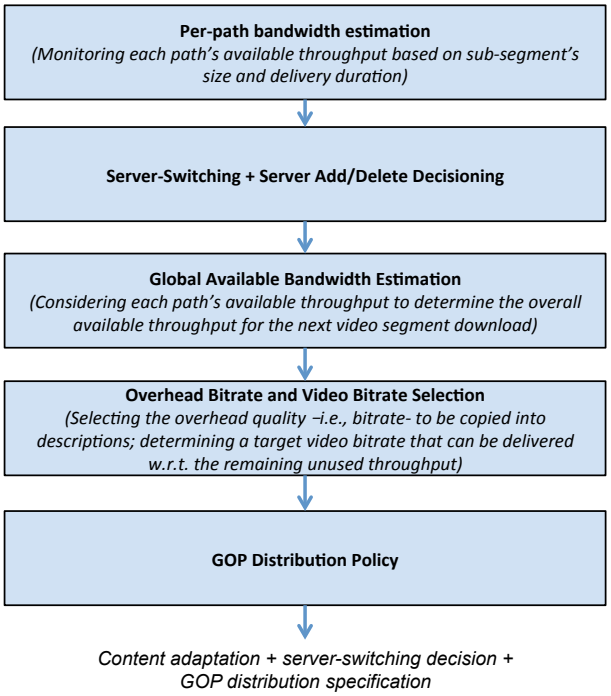

Figure 4: MS-Stream decision diagram

Flexible GoP distribution for adaptability to heterogeneous network characteristics

HTTP-based video streaming in environments where resource heterogeneity can be significantly wide is a technical challenge that can be addressed via adaptability in the amount of requested throughput on each path. The GoP distribution step is responsible for designing descriptions requests (i.e., determining their composition in terms of GoP at a selected *B* bitrate) so as to map their bitrate to the estimated throughput on each path.

Based on the listed servers and on the GoP information from the manifest file, the MS-Stream client specifies the GoP composition of description segments that each server should deliver. More explicitly, the MS-Stream client requests (through parameters in the HTTP request) each server to generate a segment with some GoPs at a selected *B* bitrate and the remaining ones at a critically-low *L* bitrate. Let us consider a video segment with *N* GoPs that the MS-Stream client wishes to display at bitrate *B* while considering the use of *S* servers simultaneously. Let us define $GoP_{i,B,s}$ so that $GoP_{i,B,s} = 1$ when server *s* is assigned the delivery of GoP *i* at bitrate *B* and $GoP_{i,B,s} = 0$ otherwise. The MS-Stream client requests each server to deliver a playable description segment so that equation (1) and (2) are satisfied:

$$\sum_{i=1}^{N} GoP_{i,B,s} \leq N \qquad (1)$$

$$\sum_{s=1}^{S} \sum_{i=1}^{N} GoP_{i,B,s} = N \qquad (2)$$

under the constraint that the delivery of GoP *i* in the targeted *B* bitrate is assigned to one server only. Thus, servers are being assigned the generation and delivery of segments with different GoPs at the selected high bitrate. Eventually, the MS-Stream client can reconstruct a segment at the requested *B* bitrate.

Content adaptation for QoE enhancement and bandwidth overhead reduction

As in the DASH standard, the MS-Stream protocol has adaptation capabilities. Given a video segment, as soon as all descriptions are delivered or some downloads cancelled, the client estimates the status of each used path (by measuring the throughput), and computes the global available throughput. Then a candidate bitrate B is selected for the next segment and each server is requested the delivery of descriptions.

As in DASH, our proposal is agnostic to the implemented bitrate adaptation methods/algorithms, which makes it highly flexible and adapted to existing or future approach. The main objective of our contribution is not on optimal bitrate adaptation methods, but to propose, explain and demonstrate that bitrate adaptation is feasible in our client-centric approach for the simultaneous usage of multiple heterogeneous servers.

MS-Stream also addresses the issue of minimizing bandwidth overhead coming from the multiple-description coding approach. Bandwidth overhead is defined as the percentage of data transiting on the network, which does not take part into the displayed content. We have $PlayedData + NonPlayedData = TransmittedData$, therefore:

$$Overhead = 1 - \frac{PlayedData}{TransmittedData} \qquad (3)$$

The capacity of a streaming system to accommodate as many clients as possible is function of its global upload throughput. This upload throughput capacity is linearly decreased by the bandwidth consumption overhead, hence impacting the infrastructure scalability, which in turn impacts on consumer's QoE. Minimizing this overhead is therefore an essential issue. Consequently, when a steady state is reached in a streaming session where descriptions are always being received in time, MS-Stream clients may modify segment requests and lower the amount of redundancy per description (down to an almost null overhead). Moreover, unless servers are specified not to include any redundancy in the generated sub-streams, the bandwidth overhead increases with the amount of used servers. Note that this present work does not investigate behaviors to be adopted by MS-Stream clients in order to optimally deal with the number of simultaneously used servers and overhead thresholds.

Server adaptation for a flexible usage of network resources

As above-mentioned, a manifest file containing information on servers' availability is delivered to the client from the resources provider. This manifest is then periodically updated to let resources providers adapt the pool of available servers. Since the retrieved descriptions are independent from each other, the MS-Stream protocol also offers clients opportunities to seamlessly change the number of considered servers during streaming sessions (by adding, removing servers). Several types of server adaptation policies can be implemented with regards to observations on servers' characteristics (available throughput, delay) and to QoE objectives (a targeted percentage of the video duration in a requested bitrate with limited rebuffering events). For example, priority could be given to visual content quality stability by switching to or adding more stable servers to the session when the considered ones are very unstable and are assigned an important number of GoPs. MS-Stream also supports the use of a large range of different content source types such as a set of Set-Top-Boxes and cloud servers. Finally, when the MS-Stream client switches to the use of one server, a regular DASH session takes places.

3 Evaluations

3.1 Evaluation test-bed

In controlled environment

Our evaluation benchmark is composed of six different ways of streaming video content, including three fully implemented applications based on dash.js player (*http://dashif.org*): Multi-Source Non-Adaptive streaming, MS-Stream-BA with Bitrate Adaptation only and MS-Stream with bitrate and server adaptation.

The three others have been designed to represent the full potential of both DASH and MS-Stream in uni-source and multi-source environments. These clients foresee the optimum of each solution. Referred as "oracle" clients, they know in advance all upcoming throughput variations. Oracles maximize QoE by retrieving the highest possible content bitrate while minimizing video freezing events. They can be considered as the optimal solutions of (1) DASH (Uni-Source-DASH-oracle), (2) Multi-Source-DASH (Multi-Source-DASH-oracle) (3) MS-Stream (MS-Stream-oracle). In our study oracles are used as reference use cases, to allow QoE comparison. Table 1 provides details on the evaluated applications, their consumption and adaptation capabilities.

Table 1: Test-bed applications

Applications	Available Servers	Simultaneously used servers	Consumption styles and adaptation capabilities
Uni-Source-DASH-oracle	3	1	Bitrate + Server adaptation *Always downloading from the best path* *based on prior knowledge of throughput fluctuations*
Multi-Source Non-Adaptive	3	3	Fixed GoP distribution policy (4 GoPs out of 12 in 6Mbs –at *B* bitrate- and 8 GoPs out of 12 in 200Kbps –at *L* bitrate) from each server)
MS-Stream-BA	3	3	Flexible GoP distribution policy + Bitrate adaptation *based on throughput estimation*
MS-Stream	9	3	Flexible GoP distribution policy + Bitrate + Server adaptation *Switching from the less-performing server to another randomly selected server every new segment download only if the GoP distribution step assigns a given server with 10 or more GoPs out of 12, based on throughput estimation*
MS-Stream-oracle	9	3	Flexible GoP distribution policy + Bitrate + Server adaptation *based on prior knowledge of throughput fluctuations*
Multi-Source-DASH-oracle	9	3	Bitrate + Server adaptation *Downloading the first segment on the best path and the two next segments on the two next best paths* *based on prior knowledge of throughput fluctuations*
dash.js Internet	1	1	Bitrate adaptation as implemented in the dash.js client from DASH-IF
MS-Stream Internet	9	3	Flexible GoP distribution policy + Bitrate + Server adaptation *Switching from the less-performing server to another randomly selected server every new segment download only if the GoP distribution step assigns a given server with 10 or more GoPs out of 12, based on throughput estimation*

In this evaluation, we consider the perceived quality at the consumer's side, i.e., QoE. To this end, we have derived a set of criteria (each considered essential for video streaming services' QoE): quality distribution throughout the streaming session, average number of quality changes, mean displayed bitrate, average number of rebuffering and average start-up delay. We also evaluated the bandwidth and CPU overhead of our solution.

The 10-minute Big Buck Bunny movie was encoded at 7 different bitrates in H.264 (200Kbps, 1Mbps, 1.5Mbps, 2Mbps, 3Mbps, 4Mbps and 6Mbps) and was segmented in 6-second segments, each containing 12 GoPs. The Multi-Source Non-Adaptive and MS-Stream clients used the 200Kbps quality as the redundancy to be copied into requested segments.

The DASH Industry Forum provides benchmarks for various aspects of the DASH standard [6] including six Network Profiles (NPs) featuring different bandwidths, delays. Each profile spends 30 seconds for each step described in Table 2, then starts back at the beginning. For each streaming session, the channels between the client and the servers are distinct. A specific NP is associated to all channels. A random time offset is set to each assigned NP so as to represent bandwidth diversity and variability in the network. We repeated each video 50 times per application and per NP; a total playback time of 150 hours was performed.

Table 2: Network Profiles representing throughputs and delays

#1 Mbps (ms)	#2 Mbps (ms)	#3 Mbps (ms)	#4 Mbps (ms)	#5 (Mbps, ms)	#6 Mbps (ms)
5.0 (38)	5.0 (13)	5.0 (11)			
4.0 (50)	4.0 (18)	4.0 (13)	9.0 (25)	9.0 (10)	9.0 (6)
3.0 (75)	3.0 (28)	3.0 (15)	4.0 (50)	4.0 (50)	4.0 (13)
2.0 (88)	2.0 (58)	2.0 (20)	2.0 (75)	2.0 (150)	2.0 (20)
1.5 (100)	1.5 (200)	1.5 (25)	1.0 (100)	1.0 (200)	1.0 (25)
2.0 (88)	2.0 (58)	2.0 (20)	2.0 (75)	2.0 (150)	2.0 (20)
3.0 (75)	3.0 (28)	3.0 (15)	4.0 (50)	4.0 (50)	4.0 (13)
4.0 (50)	4.0 (18)	4.0 (13)			

Over the Internet

We conducted experiments in uncontrolled networks over the Internet on our demo website[4]. For 3 days, 5 hours per day, 20 concurrent *"MS-Stream Internet"* clients located in 4 different cities (Paris, Bordeaux, Warsaw and Bucharest) watched videos hosted on different type of servers located in the latter four cities, having various upload throughput capacities: 9 set-top-boxes at 5Mbps, 2 set-top-boxes at 20Mbps and 1 cloud servers at 20Mbps. Each MS-Stream client was simultaneously using 3 randomly assigned servers and had 9 available servers in total. For 3 other days, 5 hours per day, 20 concurrent uni-source *"dash.js Internet"* client were consuming content from the same randomly assigned servers. As in section 3.1, we evaluated QoE metrics and overheads.

3.2 Evaluation results

QoE evaluation

1) Rebuffering per session (Figure 5.a) and average start-up delay: For all applications except Multi-Source Non-Adaptive, rebuffering almost never took place (< 0.1 freezing events per streaming session). For the Multi-Source Non-Adaptive client, rebuffering occurred between 0.12 and 1.22 times on NPs #1-#2-#3 and between 2.76 and 4.08 times on NPs #4-#5-#6. As to the MS-Stream Internet clients, few rebufferings occurred (1.26) compared to dash.js Internet (3.42), thanks to redundancy copied into sub-segments.

As MS-Stream-BA and MS-Stream solutions played the first segment as soon as they retrieved a description (and immediately canceled the other description downloads), they had the lowest average start-up delay (1.10 seconds on NPs #1-#2-#3 and 0.87 seconds on NPs #4-#5-#6). Uni-Source-DASH-oracle and Multi-Source-DASH-oracle had longer delays (respectively 2.03 seconds and 1.45 seconds on NPs #1-#2-#3 and 1.81 seconds and 0.93 seconds on NPs #4-#5-#6) compared to MS-Stream.

2) Mean Bitrate (Figure 5.b): Uni-Source-DASH-oracle reached a mean bitrate of approximately 3385Kbps on NPs #1-#2-#3 and 4416Kbps on NPs #4-#5-#6. Using multiple servers simultaneously, Multi-Source Non-Adaptive

streaming achieved higher bitrate on NPs #1-#2-#3 with an average 43.3% increase and 6.1% on NPs #4-#5-#6. Adaptive GoP distribution policy and bitrate adaptation enabled a better usage of network resources and increased MS-Stream-BA's mean bitrate to approximately 5733Kbps on NPs #1-#2-#3 and 5540Kbps on NPs #4-#5-#6. Finally, leveraging the protocol with server adaptation (switching) increased the mean bitrate in MS-Stream to 5900Kbps on NPs #1-#2-#3 and 5845Kbps on NPs #4-#5-#6, respectively representing 74% and 32% gains compared to Uni-Source-DASH-oracle. MS-Stream results are almost similar to the multiple-source oracle clients (Multi-Source-DASH and MS-Stream-oracle).

Dash.js Internet clients only attained a 2.82Mbps mean bitrate whereas MS-Stream Internet performed much better (5.28Mbps).

3) Content quality distribution (Figure 5.c) and number of quality changes: Multi-Source Non-Adaptive streaming displayed the top content quality for approximately 78% of the time on all NPs, displaying the 200Kbps quality the rest of the time. The Multi-Source Non-Adaptive streaming client frequently had to abandon descriptions download

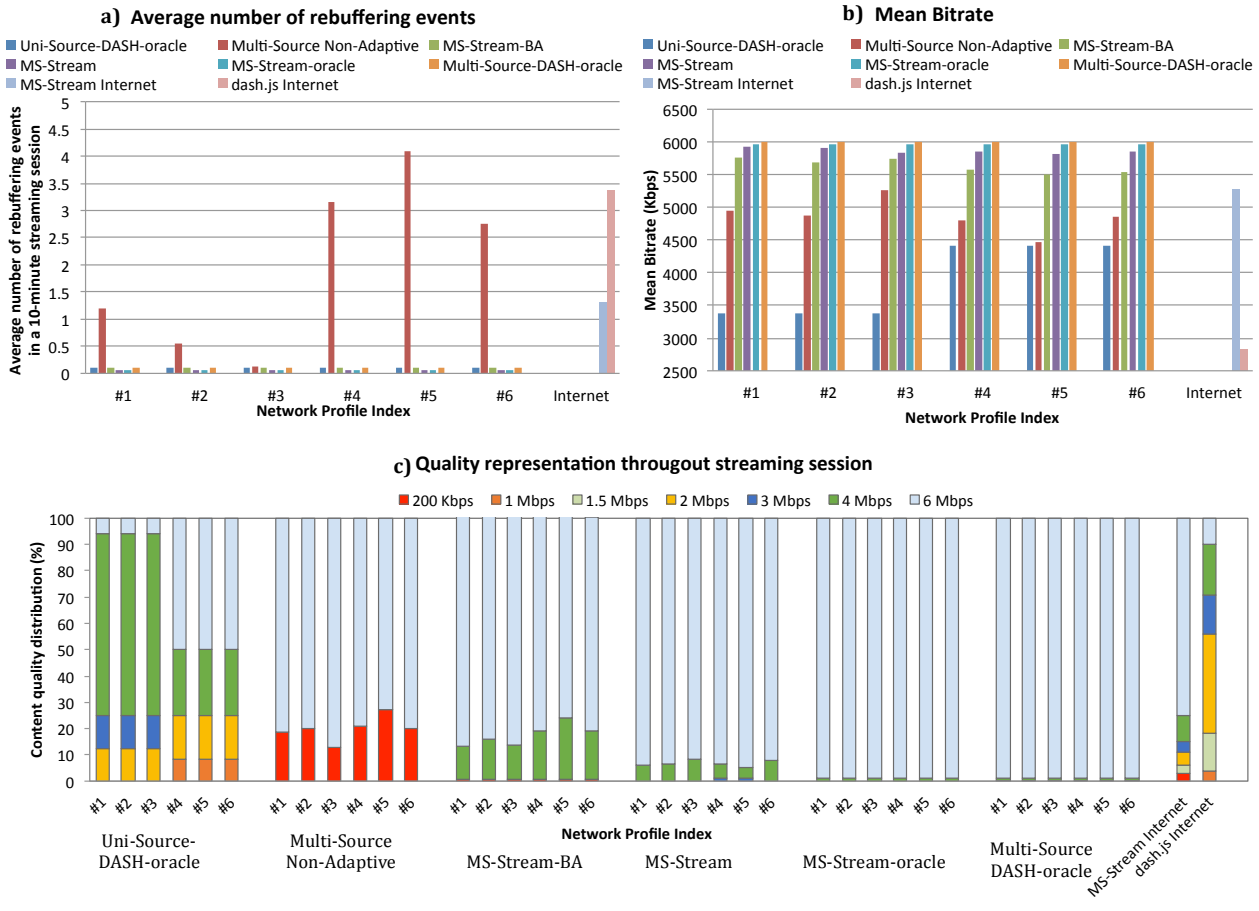

Figure 5: QoE comparison between DASH and MS-Stream

due to network heterogeneity, resulting in many quality fluctuations (between 27 and 32 changes on NPs #1-#2-#3, 39 and 52 on NPs #4-#5-#6) from 6Mbps to 200Kbps. The benefits of MS-Stream-BA are visible as the 200Kbps displayed during for 22% of the time in the Multi-Source Non-Adaptive experiments could be transformed into 4 Mbps for 17% of the time for MS-Stream-BA. Additionally, quality changes occurred seldom for MS-Stream-BA compared to Multi-Source Non-Adaptive streaming (less than 5 on all NPs).

Relying on server adaptation, MS-Stream took benefit of greater link diversity and eventually provided the 6Mbps

quality for 93% of the time in average on all NPs with very few quality changes (less than 3 on all NPs). For the Uni-Source-DASH-oracle client, although 3 servers were available and Uni-Source-DASH-oracle had prior knowledge of upcoming throughput fluctuations, the use of one server only was not enough to provide even a similar quality distribution compared to MS-Stream as shown in Figure 5.c. MS-Stream-oracle and Multi-Source-DASH-oracle performed similarly (99% of 6Mbps during streaming sessions).

At last, MS-Stream Internet greatly outperformed dash.js Internet clients in terms of quality fluctuations (9.28 versus 27.2) and distribution (75.3% versus 9.9% of the time at 6Mbps).

Overhead evaluation

1) Bandwidth consumption overhead(Figure 6.a): On all NPs, multiple-description-based applications presented an overhead lower than 7%. Interestingly, Multi-Source Non-Adaptive had the least amount of overhead (6.07%) due to the fact that many description downloads were cancelled; consequently displaying the copied redundancy from the received description segments. For their parts, MS-Stream-BA and MS-Stream had an overhead near 6.5% on all NPs, which is very close to MS-Stream-oracle's overhead (6.25%). Finally, the bandwidth overhead of MS-Stream Internet clients rose up to 7.18%, higher than during the evaluations on NPs due to a lower mean bitrate and quality distribution.

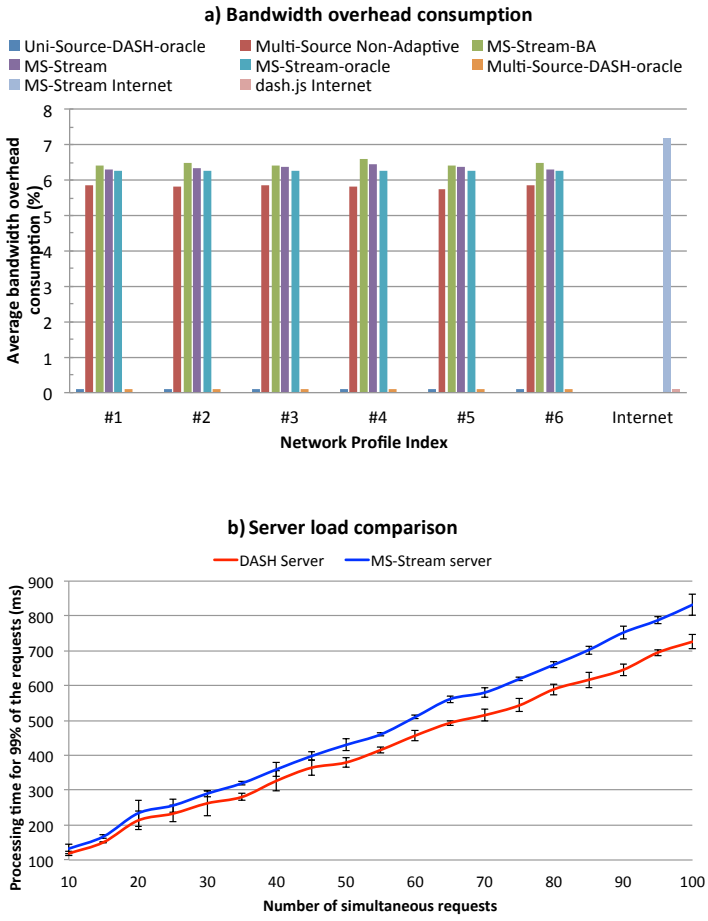

Figure 6: Overhead evaluation

2) Server CPU overhead(Figure 6.b): Regarding the impact of description generation on server, we compared a DASH server and a MS-Stream server based on the processing time required to handle simultaneous requests (normal segments for DASH; description creation with random GoP composition for MS-Stream). Both servers were implemented in Java. Each server was required to handle 10000 requests with a number of simultaneous requests indicated in the x-axis of Figure 6.b. Segment bitrates were randomly selected between 500Kbps and 6Mbps with 500Kbps steps. The redundancy was set to 200Kbps. We could show that in 99% of the cases, the processing time for a MS-Stream request is 11% greater than for a DASH request.

In a nutshell, we demonstrated that MS-Stream has the potential to deliver higher QoE than DASH with significant mean bitrate increase, less rebuffering, shorter start-up delays and less quality fluctuations at the cost of light overheads in terms of bandwidth consumption (6.5%) and CPU.

4 RELATED WORK

HAS protocols have seen important interest from the industry and the research community, mainly due to their capabilities to render smooth video playback to the consumers, hence a better QoE. The authors in [3] and [7] introduced the DASH framework of Netflix, and outlined that a user is always bound to one server, regardless of the available throughput between the user and the server. [3] also indicates that QoE could greatly benefit from the venue of a practical HAS that can actually utilize multiple servers simultaneously. In [8], the proposed bitrate adaptation decisions are asynchronously taken, imposing segments to be retrieved one after another without considering completion time. The contribution in [10] presents Presto, a streaming protocol designed to use several servers simultaneously to improve QoE by providing better fairness, efficiency and stability at server side. However, Presto performs very poorly should any of the considered network paths suffer degradation. Adopting [8] and [9] for multiple-source streaming will result in requests being most of the time delivered out-of-date due to path heterogeneity, leading to video rebuffering. In contrast, our proposed solution considers requesting several servers simultaneously and abandoning segment requests, which cannot meet their assigned deadline. In [10], an evolution of DASH is put forward using multiple servers with Scalable Video Coding techniques where content is generated over a ”single base layer” providing a minimum visual quality, and several dependent ”enhanced layers” increasing the rendered quality. Failing in retrieving the appropriate layers will prevent the consumer from watching the video. In contrast with [9] and [10], our sub-stream generation scheme relies on the principles brought by Multiple Description Coding (MDC[5]) to create independent and aggregatable sub-segments, thus providing uninterrupted streaming even when segments are lost. Although most of the proposed MDC approaches[5] rely on two descriptions only, our solution can generate any number, enabling the use of an important number of servers, providing better bitrate scalability and resilience to outages and delay variations[5]. The work [11-12] also relies on the use of descriptions. However, descriptions in [[11-12] are created prior the content consumption phase that limits network heterogeneity adaptability to the number of available descriptions, leading to many quality fluctuations. Moreover, description placement in a distributed system is an obvious issue of [11-12], requiring expanded storage capacity. In our work, descriptions are generated for each video segment, during consumption phase, and upon client specification. There are other multiple-source streaming approaches. However they are based either on streaming consecutive pieces of content as a playlist or on switching to another server when HTTP servers response with error codes, as explained in the DASH standard [13] or exposed in [3].

Another approach in multi-path streaming is the use of the multiple-path TCP protocol (MPTCP). Many studies have shown that MPTCP performances are below expectations, especially under heterogeneous network characteristics [14]. In MPTCP, a scheduler assigns each packet from the MPTCP output queue to one available TCP connection. Packet losses occurring on one path can generate head-of-line blocking at client side leading to fast buffer depletion. Many papers addressed this issue by focusing on windows congestion management [15], cross-layer scheduling [14], bandwidth and buffer management [16] and retransmission [17] at the transport layer. Our approach differs from MPTCP streaming by performing both multi-source/multi-path delivery and adaptation at the application layer, using

TCP as its transport mode. Furthermore, by benefiting from the non-dependency between sub-streams, MS-Stream provides an elegant client-centric way to handle stream synchronization from multiple sources. Finally, while multi-homing approaches [18][19] use multiple access networks to connect to one server and improve network goodput, they are limited by the number of available access networks. P2P streaming represents another multi-source streaming solution, however the challenges of P2P streaming lie in its vulnerability (peer churn, imbalance capacities, flash crowds). Our approach takes a different perspective by simultaneously using multiple servers regardless of the type of used networks in order to provide smooth streaming by adapting the video bitrate quality.

5 CONCLUSION

We presented an evolving streaming solution to pragmatically enable HAS in multiple-source environments, providing significantly QoE enhancements. Resiliency and adaptability to network heterogeneity were addressed by simultaneously retrieving independent sub-streams from several servers generated upon clients' specifications. A flexible usage of network resources is proposed by adapting content bitrate and used servers. An online demonstration of our work is available [4].

In short term, the adoption of MS-Stream-like solutions could allow resources providers (ISPs, IoT, cloud, set-top-box providers) to be involved in the content delivery value chain. By using their horizontally scalable distributed infrastructures, they could sell content delivery services over the Internet at cheaper prices compared to current CDNs, and with similar, if not higher QoE capabilities. Additionally, with MS-Stream's capacity to use any type of content source in any distributed infrastructure, resource providers could dynamically adjust the scale of their infrastructure in order to meet audience dynamicity and QoE demand increase.

REFERENCES

[1] E. Baccaglini, "A study of an hybrid CDN-P2P system over the planetlab network," *Signal Processing: Image Communications*, 2012.

[2] D. Xu, S. S. Kulkarni, C. Rosenberg, and H.-K. Chai, "Analysis of CDN-P2P hybrid architecture for cost-effective streaming media distribution," *Multimedia Systems*, 2006.

[3] V. K. Adhikari et al., “Unreeling Netflix: Understanding and improving multi-CDN movie delivery,” *IEEE INFOCOM,* 2012.

[4] J. Bruneau-Queyreix, et al., “A multiple-source adaptive streaming solution enhancing consumer's perceived quality," in *IEEE Consumer Communications and Networking Conference (CCNC) - demonstration track* 2017, accessible online: http://msstream.net

[5] M. Kazemi, S. Shirmohammadi, and K. H. Sadeghi, “A review of multiple description coding techniques for error-resilient video delivery,” *Multimedia Systems*, 2013.

[6] “Guidelines for implementation: DASH-AVC/264 Test cases and Vectors,” http://dashif.org/wp-content/uploads/2015/04/DASH-AVC-264-Test-Vectors-v09-CommunityReview.pdf

[7] M. Watson, “HTTP Adaptive Streaming in practice,” http://web.cs.wpi.edu/claypool/mmsys-2011/Keynote02.pdf, last visited: Feb 2016.

[8] G. Tian and Y. Liu, “Towards Agile and Smooth Video Adaptation in HTTP Adaptive Streaming,” IEEE/ACM Transactions on Networking, 2016.

[9] S. Zhang, B. Li, and B. Li, “Presto: Towards fair and efficient HTTP adaptive streaming from multiple servers,” *IEEE International Conference on Communications (ICC)*, 2015.

[10] W. Pu, Z. Zou, and C. W. Chen, “Dynamic Adaptive Streaming over HTTP from Multiple Content Distribution Servers,” *IEEE Global Telecommunications Conference (GLOBECOM),* 2011.

[11] J. Bruneau-Queyreix, et al., “MS-Stream: A multiple-source adaptive streaming solution enhancing consumer's perceived quality," in *IEEE Consumer Communications and Networking Conference (CCNC),* 2017.

[12] J. Bruneau-Queyreix, et al., “Multiple description-DASH: Pragmatic video streaming maximizing end-users' quality of experience." *IEEE International Conference on Communications (ICC)*, 2016

[13] ISO/IEC 23009-1:2014 - Information technology - Dynamic adaptive streaming over HTTP (DASH).

[14] X. Corbillon, et al., “Cross-layer scheduler for video streaming over MPTCP,” ACM *International Conference on Multimedia Systems (MMSys)*, 2016.

[15] N. Kuhn, et al., “DAPS: Intelligent delay-aware packet scheduling for multipath transport,” *IEEE International Conference on Communications (ICC)*, 2014.

[16] T. Kurosaka and M. Bandai, “Multipath TCP with multiple ACKs for heterogeneous communication links,” *IEEE Consumer Communications and Networking Conference (CCNC)*, 2015.

[17] C. Raiciu et al., “How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP,” *USENIX Symposium on Networked Systems Design and Implementation (NSDI),* 2012.

[18] Y. Go et al, "An Energy-Efficient HTTP Adaptive Video Streaming With Networking Cost Constraint Over Heterogeneous Wireless Networks", *IEEE Transactions on Multimedia*, 2015.

[19] Bui et al. "GreenBag: Energy-Efficient Bandwidth Aggregation for Real-Time Streaming in Heterogeneous Mobile Wireless Networks", IEEE *Real-Time Systems Symposium*, 2013.

Biography

Joachim Bruneau-Queyreix received his M.Sc in telecommunications from ENSEIRB-MATMECA Bordeaux graduate school of engineering. In 2014, he began his Ph.D at University of Bordeaux in the field of multi-criteria optimization for content delivery. His research focuses on multimedia streaming, network systems and architectures, 5G, SDN/NFV. He has joined 3 European research projects in the field of ICT (FP7, CHIST-ERA, EUREKA). Contact him at jbruneau@labri.fr or jbruneauqueyreix@viotech.net

Mathias Lacaud received his M.Sc. in telecommunications at ENSEIRB-MATMECA graduate school of engineering, Bordeaux, in 2016. He worked as a research engineer in the field of optimization for multimedia content delivery for one year. Since 2017, he is pursuing his Ph.D. at LaBRI/Bordeaux Computer Science Laboratory on delivering and securing social multimedia contents over the Internet of Things. Contact him at mlacaud@viotech.net

Daniel Négru received his Ph.D from the University of Saint Quentin en Yvelines. In 2007, he became associate professor at ENSEIRB-MATEMCA/University of Bordeaux. His research interests include multimedia and networking. From 2010 to 2014, he coordinated the FP7 ALICANTE project tackling networking and multimedia research fields. He participated in more than 10 collaborative research projects, published more than 50 papers. Contact him at daniel.negru@labri.fr

Jordi Mongay Batalla received Ph.D. degree from Warsaw University of Technology (Poland). He is with Warsaw University of Technology and with National Institute of Telecommunications, where he is Head of Internet Technologies and Applications Department. His research focuses mainly on Technologies (4G and 5G, Network

services chaining, SDN) and Applications (Internet of Things, Smart Cities, multimedia) for the Future Internet. Contact him at jordim@interfree.it

Eugen Borcoci is full professor at Electronics, Telecommunications, and Information Technology Faculty of University Politehnica Bucharest (UPB), Romania. His expertise comprises theoretical and experimental knowledge in telecommunication/computer systems. His recent areas of interest are future Internet architectures, SDN, NFC, 5G and vehicular communication technologies. He participated as team leader and researcher in many FP5, FP6, FP7, and Chist-Era European projects. Contact him at eugen.borcoci@elcom.pub.ro