SCTE CABLE-TEC EXPO® 2010 | TECHNICAL PAPER

The Case for Adaptive Streaming

Asaf Atzmon, Sr. Director at the CTO Office
BigBand Networks
475 Broadway Street, Redwood City, CA 94063
Phone: 617-538-2457 | Asaf.atzmon@bigbandnet.com

Introduction

Innovations in the space of online video delivery are gaining popularity as over the top (OTT) providers are looking for ways to improve the Quality of Experience (QoE) for their viewers. The industry has recently seen the popularization of several methods of adaptive streaming from vendors such as Apple, Microsoft and Adobe. These methods claim to provide a better user experience by enabling video-optimized adaptations when faced with dynamic changes in network conditions. We expect these methods to become a de-facto standard for delivering OTT media through the massive deployment of compliant video players such as Adobe Flash, Microsoft Silverlight and Apple QuickTime.

As MSOs are looking to expand their TV and on-demand services to reach IP-connected devices, they may be inclined to adopt such methods as a quick way to deliver an adequate quality of experience to PCs and other devices. However, doing so does not take full advantage of the ability to manage network resources. Conversely, in adopting a hybrid view for adaptive streaming that includes aspects of session management and dynamic network resources allocation, MSOs can provide a differentiated value for their IP target services as well as enable more efficient usage of their networks.

The issues around bandwidth utilization, quality of service and user experience in video delivery over managed and unmanaged networks is a common research area with well-documented academic and commercial approaches. This paper will review the ways in which adaptive streaming can address these issues, and the market and technology drivers and different technologies that support it.

Specifically, this paper will discuss the technology behind popular methods of adaptive streaming and will position them in the context of solutions for rate control and congestion avoidance. It will argue that the current methods in place today were designed with an end-to-end mechanism in mind that abstracts the communication network between the sender and the receiver and perceives it as a ‘hostile’ element. It will then discuss the general principles of bandwidth management and resources allocation in a managed network and the problems they aim to solve. Next, the paper will point to some inconsistencies that common adaptive streaming methods have with these principles. Lastly, the paper will suggest a hybrid view that includes the network elements as a cooperative function to the end-to-end mechanism in a way which would further optimize users' QoE and network resources allocation.

Over the Top Video

Recent years in media entertainment have bared witness to the massive decline in broadband costs, which has enabled the democratization of content production and distribution in what has come to be known as OTT delivery. OTT uses standard Internet protocols to deliver audio-visual content over any IP-capable network The advent of mobile devices coupled with the introduction of Wi-Fi and 3G capable smart-phones, such as the iPhone and DROID, along with the increasing connectivity of home entertainment devices, including the TV set, have created a foundation to expand the consumption of OTT content beyond the PC screen. The term OTT alludes to the fact that this content is typically delivered to end users **unmanaged** over the access network. This means that from the perspective of the network provider, OTT traffic is regarded as High-Speed Data (HSD) which receives a best-effort Quality of Service (QoS) treatment.

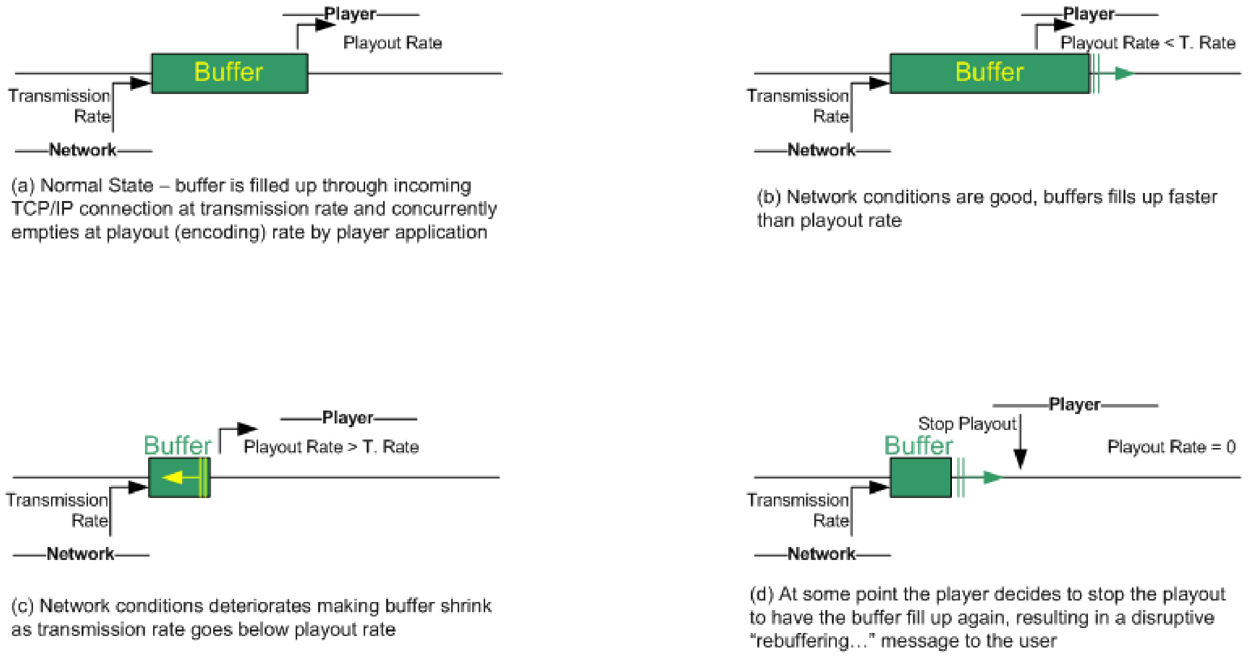
The technical foundations of OTT delivery systems are grounded in the history of online media streaming which has its roots back to the early days of the World Wide Web circa 1995 with Microsoft, RealNetworks and Apple as the leading vendors for online media servers. At a later point, with the growing popularity of the Adobe Flash player and its inherent video capabilities it became the most popular online video player used across a majority of today's online video destination sites including YouTube and Hulu.

While much innovation has been introduced by vendors with respect to optimizing the delivery of streaming media to end users, this has typically been done through all sorts of proprietary mechanisms that have privately defined the syntax of the A/V payload, the traffic container and the wired protocol between the server and the client. To realize these benefits, content owners were required to put significant investment in buying specialized servers from a single vendor and then adopting a client distribution strategy in hopes of achieving a significant footprint of users that could view their content. Given Internet users' reluctance to download custom clients with no clear method to authenticate trust, only the biggest vendors were able to successfully leverage a substantial footprint on the client side. Those companies included Microsoft with their hold over the Windows OS, Apple with their Mac, and more importantly iPod/iPhone/iPad dominance and Adobe, who leveraged their Flash success. RealNetworks, once a lead player, saw its market share evaporate much as many smaller startups went up and down the media players' rollercoaster.

The Role of Progressive Download

Even with support from the biggest vendors, the cost of deploying specialized media servers remained an impediment for many of the content providers who were looking for ways to utilize off-the-shelf hardware and commodity software to deliver their streams.

These drivers have created a surge of deployments of Hypertext Transfer Protocol (HTTP)-based web servers for media delivery in what was to become known as progressive download. Progressive download uses off-the-shelf web servers such as Apache to host and deliver media content over HTTP in a complete standard way. In other words, from the server perspective - HTTP GET requests are received from the client for a specific piece of content and the server uses its standard HTTP and Transmission Control Protocol/Internet Protocol (TCP/IP) mechanism to fetch back the content bytes. The aspect of progressive in this method stems from the client ability to fill up buffers and play content off them simultaneously. To that end, the client employs some internal mechanism to infer what is an adequate buffer fullness. This allows the playout of content with low risk of buffer underflow. However, since the client (as well as the server) has no control over the network performance over the period of the session, it is limited in its ability to accurately predict these measures, which in turn could cause the undesirable phenomena of "rebuffering" (the client halts playout to reach a higher buffer fullness). Figure 1 demonstrates progressive download operation:



(a) Normal State – buffer is filled up through incoming TCP/IP connection at transmission rate and concurrently empties at playout (encoding) rate by player application

(b) Network conditions are good, buffers fills up faster than playout rate

(c) Network conditions deteriorates making buffer shrink as transmission rate goes below playout rate

(d) At some point the player decides to stop the playout to have the buffer fill up again, resulting in a disruptive "rebuffering…" message to the user

Figure 1: Progressive Download Operation

Progressive download provides several benefits over previous proprietary methods:

1. As previously mentioned, the overall cost of deploying off-the-shelf web servers is significantly lower than that of deploying specialized proprietary streaming servers.
2. Since progressive download uses HTTP at the application/transport layer, it transverses corporate firewalls and other security gateways which often employ restrictions on other types of traffic such as User Datagram Protocol (UDP) or the proprietary Multimedia Messaging Service (MMS) and Real

Time Messaging Protocol (RTMP) that are typically used with streaming media servers.

3. By default, HTTP traffic is cacheable on downstream cache nodes. This provides great scale benefits for content providers that can leverage ISP caching to offload a big part of the demand on their servers. It also allows Content Delivery Network (CDN) vendors to provide low-cost HTTP caching to their customers instead of deploying a massive capacity of custom media servers.

Progressive download initially had some limitations related to HTTP's inability to support random access, which means that users were given the option to scan across the content only along the boundaries of what has already been buffered by the end device. This has been addressed with the introduction of HTTP 1.1., which allows the client to include a *Range* specifier in its request, indicating the *bytes* offset for the desired resource.

All these advantages have contributed to the fact that the majority of online video traffic is currently delivered using progressive download. Only real-time content (news, sports) still requires the use of customized media servers.

Flow Control and Congestion in OTT Media

With all of the advantages of using HTTP servers for the delivery of online video, there are some aspects in which the approach falls short when compared to proprietary streaming servers. One of these aspects is the HTTP server's inability to provide any mechanism for flow control of the session between the server and the client. In simple terms, HTTP progressive download provides **no** flow control in the application level. That's not to say there is no flow control at all. Since HTTP is using TCP/IP at its transport layer, progressive download traffic adheres to the TCP/IP sliding window mechanism where the client and server negotiate a transmission rate that both can process and that, all else being equal, will quickly stabilize around the maximum value that can be allowed regardless of what the *actual* bitrate is of the content being delivered. For that reason, progressive download traffic typically demonstrates a *greedy* behavior filling up client buffers as fast as possible. This greediness may in fact have at times some positive impact on the user QoE as it reduces the client's dependency on the server or the network for the remainder of the session. On the flip side, this greediness could cause a lot of wasteful traffic that has to be accounted for by the network provider, including all the data that has been downloaded but never actually watched by the end user.

Another property of TCP/IP traffic is congestion avoidance. As an end-to-end protocol, TCP/IP provides no visibility into the congestion level of the network links that carry the traffic. Some mechanisms are inherent in the protocol level to demonstrate "good citizenship" and protection against a 'congestion collapse'. A 'congestion collapse' refers to when a link in the network that is faced with high incoming traffic transitions into a state of discarding packets, causing senders to worsen the situation by retransmitting data and effectively causing the network to allow only a very low throughput out.

The problem of having progressive download rely solely on the underlying TCP/IP behavior to provide congestion avoidance is that it is only as efficient as any IP traffic; in other words, since it doesn't take into account the nature of the A/V payload, no efficiencies related to the video layer are gained. The results are a pretty arbitrary impact on the video QoE achieved. The good scenario would have an end client initiating a session under relatively good network condition allowing the client to effectively download the entire content in a reasonable rate, enjoying a "local" QoE. This is also beneficial from the network perspective as that content demand has been offset to the client side relaxing future demand on the network. More often than not, however, greedy clients will compete over a congested link causing a compromised QoE as well as introducing artificial demand over a distressed network link.

Some mitigation techniques have been suggested to address the flow and congestion inefficiencies of TCP/IP protocol with respect to inelastic traffic:

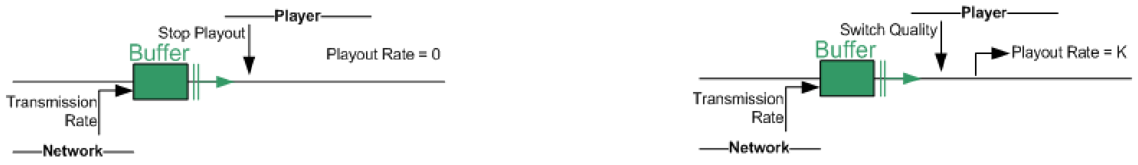
1. A self-limiting function can be introduced to the source for the server to communicate with the TCP/IP layer to achieve rate-control. The server is aware of the video encoding bitrate and can fill up its outgoing network buffers in a rate which is a factor of that bitrate. For example, the server could cap its rate per session to 20% above the content bitrate. This will provide some mitigation to the client greediness problem and will also help with the server overall scale (it should be able to support on average more concurrent sessions).
2. Adaptive streaming is another technique to mitigate TCP/IP shortcoming of rate control and QoE optimization for inelastic traffic and is the primary focus of the following section.

Adaptive Streaming in OTT HTTP Traffic

It is useful to contrast methods of adaptive streaming with that of traffic shaping. Traffic shaping is defined by Wikipedia as *"the control of computer network traffic in order to optimize or guarantee performance, improve latency and/or increase usable bandwidth by **delaying packets** that meet certain criteria".* In general, traffic shaping is applied at network links where multiple traffic flows need to be shared across a constant rate communication channel. ISPs and network providers typically employ traffic shapers as a means to cap certain IP video flows from exceeding some given transmission rates. The underlying principle is that by delaying or discarding arbitrary packets of a flow, the endpoints would interpret that as a high congestion level and would consequentially reduce the transmission rate. Depending on the encoding content bitrate and the client's buffer fullness, this overall behavior may degrade the user experience by causing delays in the delivery of frames (buffer underflow).

Similarly, adaptive streaming methods impact the transmission rate of IP video. However, instead of delaying packets in time, adaptive streaming controls the rate by dynamically changing the compression level of the A/V traffic. To fully grasp the difference in this approach, it is useful to observe the nature of the QoE tradeoff in situations where the network is congested. Rate-shaped traffic would result in the reduction of the transmission rate that at times could go below the bitrate of the encoded content. When this inequality occurs, the buffers at the client will start emptying up and at some point (based on the client's algorithm of buffer fullness) the client will halt content playout to allow his buffers to get filled again to an adequate level. This will result in a highly disruptive experience to the end user who will need to endure a discontinuity of his viewing experience.

In contrast, adaptive streaming puts the penalty of network congestion on the video quality. By switching to a lower compression level, the quality of frames will be compromised. However, the rate of frames per second delivered over the network will be kept within some fixed boundaries. As a result, the transmission rate will reduce relieving overall network congestion and, at the same time, the video quality will decrease as well, but the buffer fullness will supposedly stay relatively constant avoiding playout discontinuation. It is argued that variance in video quality is considered a less-disruptive experience than that of playout discontinuity. **That is the underlying premise behind adaptive streaming.**



(a) Progressive Download – deteriorating network conditions resulting in discontinuation of playout

(b) Adaptive Streaming – deteriorating network conditions resulting in quality switching, keeping playout rate relatively constant

Figure 2: Progressive vs. Adaptive

Some History and How Adaptive Streaming Works

Methods of adaptive streaming are hardly new in the domain of online streaming media. For commercial use of Internet video they date back as far as 1998 with the introduction of the G2 architecture by RealNetworks, which included **SureStream** technology. Around the same time, Microsoft also debuted NetShow Services 3.0 and Windows Media Player 6.1 with the **Stream Thinning** feature. These early implementations and ensuing successors all used proprietary streaming protocols to allow the control of the rate adaptation. In 2006, Move Networks pioneered the market for HTTP-based adaptive streaming. Move Network's architecture enabled the encoding of content in multiple bitrates, its fragmentation into smaller units (chunks) and the dissemination of these chunks across multiple CDNs and HTTP servers. A proprietary client would fetch the chunks and assemble them into a seamless viewing. Over the last two years, all of the leading streaming media vendors including Microsoft, Apple and Adobe have introduced products that support adaptive streaming. While each vendor provides its own specific protocols and container definitions, they all share the same underlying concepts, as follows:

- A base content that is encoded into multiple representations, each providing a different compression level with a target average bitrate;
- Per each representation, the content is fragmented into fixed duration independent units. The fragments are typically of relatively long ranging between 2 to 10 seconds;

The result of the first two steps is the formation of a matrix with time and quality/bitrate dimensions as depicted in Figure 3 below:

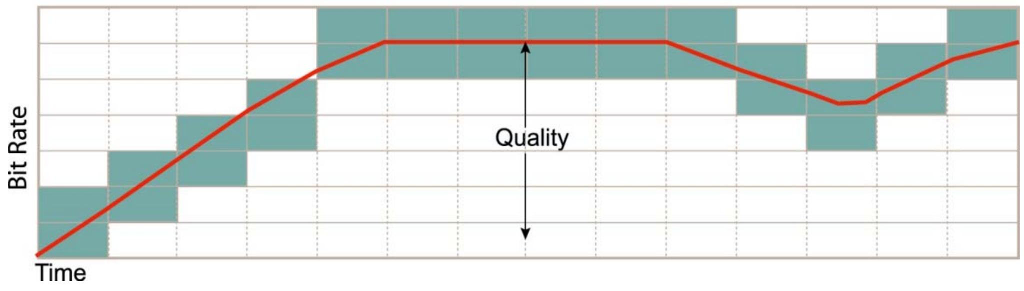


Figure 3: Time and Quality Dimensions of Adaptive Streaming

- A manifest file providing a dictionary and indices to the asset matrix.
- A client performing the following functions:
    - Continuously sends requests to the server for specific fragments. Consecutive requests will typically be for adjacent fragments across the matrix's time dimension.
    - Using an internal mechanism to observe his ability to receive and process chunks at a certain bitrate. This mechanism will be driven by certain parameters such as the incoming data rate from the TCP/IP buffers and its available processing power and memory, etc.
    - Based on the above, it may choose to switch quality, such that the next fragment it requests would be one layer up or down along the quality dimension. The process of fetching fragments across the matrix would then form a continuous curve advancing across the time dimension in a pattern like the one shown in Figure 3.

It should be noted that since the transport of fragments is HTTP-based all the advantages described earlier in this paper still hold. Some implementations currently require the use of specific web servers to fetch fragments, but this function is very likely to get standardized. Figure 4 visualizes the main elements of an adaptive streaming architecture:

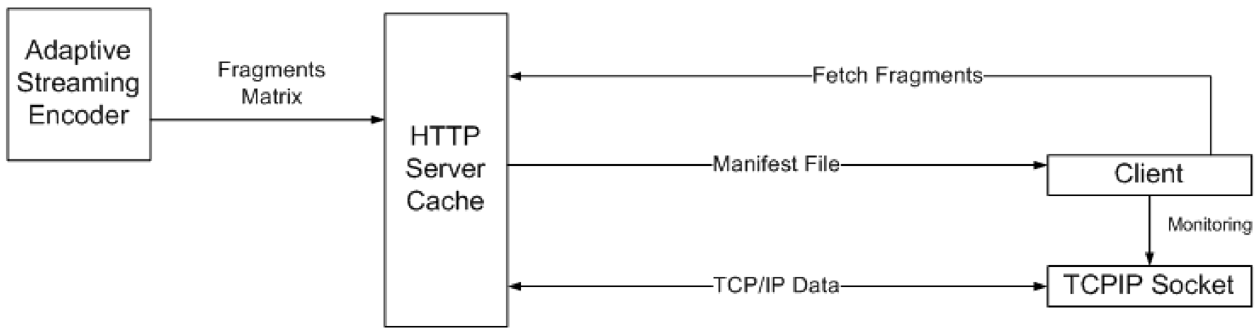


Figure 3: Adaptive Streaming Architecture

Adaptive streaming methods provide efficient means for congestion avoidance by facilitating application-level behavior that shape traffic across the quality domain rather than the time domain. To demonstrate this process, it is useful to observe the interplay between the TCP/IP layer and the application layer in the case of an adaptive streaming session.

For the sake of discussion we will assume that both server and client have infinite capacity to handle traffic so that the only potential bottleneck is the network. It is fair to assume that the client would start fetching the high quality fragments. (An implementation may choose to start with a lower quality in order to reduce start time before moving up the ladder to the higher quality). Given TCP/IP algorithms for congestion avoidance, a session will begin in a *slow-start* phase which will quickly accelerate to the maximum size agreed by the server/client before any sign of congestion is detected. It is important to note that from the application perspective, to the point where congestion is detected, the client will perceive the system is in normal health and will strive to fetch as many high-quality fragments as possible. At some point a link may get congested, in which case, the underlying mechanism will make the server significantly reduce its transmission rate; this event would signal quality degradation at the client side and will guide the client to request a lower bitrate fragment on its next request. This results in the alignment of the video bitrate with the transmission rate such that the number of A/V frames received per a unit of time is kept relatively constant. (Buffer fullness measured as a number of frames is kept above a certain low-bound threshold).

While adaptive streaming is efficient in terms of its impact on rate control and congestion avoidance, it does have some fallbacks, which include the following:

- It generates a lot of upstream traffic as each fragment fetch results in messages sent to the server. In order to reduce this effect to some extent, fragments size need to be relatively long.
- The large number of HTTP sessions has a scaling impact on the web server. In general, servers that provide video are better scaled when they have to deal with a small number of long-duration sessions.

- Due to the large size of fragments, the quality switching granularity is rather crude which puts a limit on how smooth the traffic could be shaped to address the network conditions.

To summarize, adaptive streaming methods are very efficient for avoiding congestions and improving the user's overall quality of experience. They are especially beneficial in an OTT delivery of content for the following reasons:

- They use HTTP as their transport mechanism, enabling the same benefits that made progressive download the primarily method for delivery today.
- They are designed as an end-to-end mechanism that doesn't make assumption or put dependencies on the network between the sender and the receiver. This is due to the fact that they are completely overlaid on top of TCP/IP connection and the application layer on top of it. As such, they work effectively on networks that have no built-in mechanism for QoS.

The next section will observe these aspects in the context of a managed network.

Adaptive Streaming over a Managed Network

With the advent of connected IP devices in the form of smart phones and iPads, PCs equipped with high-resolution monitors, game consoles and connected DVDs and TVs, operators are struggling to maintain their added value as the prime delivery vehicle of high-quality videos to the home over a surge of competitive OTT offering. As a response to this threat, we begin to see MSO-provided video services over their cable networks destined to IP connected devices. These services are collectively known as "TV Everywhere".

To that end, there are several alternative options that operators can choose from in terms of IP video delivery architecture:

1. Operators can adopt an end-to-end delivery approach with no QoS at the network level. This is the fastest way to deploy IP services and the approach that has currently been introduced by MSOs in their initial projects. In a non-QoS approach, operators are essentially acting as OTT providers providing only a "best effort" class of service to their users. Adaptive streaming methods would provide operators with similar benefits as was discussed in this paper. However, the obvious drawback of this approach is that it doesn't leverage MSO's managed network in providing a differentiating value over the competing offering or even as a means to better utilize resources.
2. Operators could choose to deliver the IP video services as part of the PacketCable™ architecture. The architecture defined through PacketCable Multimedia (PCMM) and the extensions introduced through PacketCable 2.0 have set the foundation for enabling real-time services to access network QoS and for the network to intelligently manage its resources allocation for sessions shared across a common network channel.
3. Operators could deliver the IP video services utilizing an Edge Resource Management (ERM)-driven architecture and edge QAM devices.

Both options (2) and (3) provide the benefits of efficient resources management and better service control due to their ability to manage *sessions*. We define a session as a state within endpoints as well as within *the network* of communication between a sender and one or multiple receivers such that network resources are allocated to support the adequate delivery of the service that session is initialized for. A*dmission control* would be the process within the network by which a session is either allowed or denied as a result of availability of network resources. PacketCable 2.0 is built upon PCMM and IP Multimedia Subsystem (IMS) to provide a Session Initiation Protocol (SIP)-centric session management with support for bandwidth management and reservation. From a technical perspective, it is well suited to support the delivery of IP video in a managed network. However, PCMM has not gained much success in today's MSO deployments beyond its use for VoIP and it is unlikely to emerge as the leading architecture to provide QoS for video services.

Conversely, the use of ERM-driven architecture is at the core of operators' narrowcast services such as Switched Digital Video (SDV) and Video on Demand (VOD) delivering high-quality video to tens of millions of digital set-top boxes. In an ERM-driven architecture, an edge QAM device provides shared bandwidth over a generally scarce RF domain. In U.S. systems, a single QAM channel translates to 38.8Mbps of bandwidth and, depending on each MSO's spectrum and architecture, a pool of QAM channels is engineered to cover a network segment or service group. Lately, operators have started demanding the use of universal QAMs that could dynamically share bandwidth across multiple services including DOCSIS® high-speed data, DOCSIS video, SDV, VOD and broadcast. At the core of this operation are the ERM element and the multiple Session Managers (SMs) that take care of session admission and bandwidth allocation.

Figure 5 shows the impact of using an ERM and a universal QAM on the utilization of the RF spectrum. As all video services are managed across the same shared pool of resources, a better utilization of network resources is achieved.

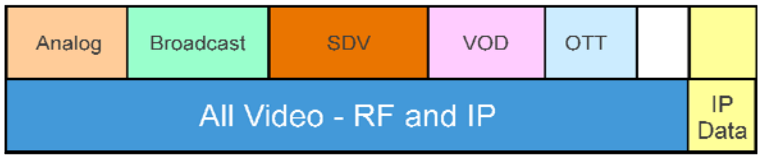


Figure 4: Universal QAM Management

Managing IP video services in ERM-centric architecture provide the operator with the following benefit:

- Ability to utilize shared resources in edge QAM devices across multiple services, including broadcast, SDV, VOD, HSD, IP video and OTT traffic.

Depending on the QAM implementation, resources can be shared at the level of a port or even a single QAM channel.

- IP video bandwidth is managed on a per session basis. Due to the fact that video sessions are typically long-lived and delivered within large sized packets, they lend themselves more efficiently to a sustained session management than to packet-by-packet QoS processing that Cable Modem Termination System (CMTS) devices typically provide for IP traffic.
- Architecture that supports edge QAMs both in a modular headend architecture (through M-CMTS) as well as over DOCSIS lite or bypass (no CMTS) configuration.

Enabling Adaptive Streaming: From Hostility to Cooperation

The case for adaptive streaming holds true for managed cable networks in much the same way as it does for OTT delivery. To recall, the objective in enabling adaptive streaming is in trading-off traffic shaping with quality switching. Having said that, the design of popular adaptive streaming methods regard the network as a 'hostile' element where no QoS can be guaranteed and, in some cases, traffic may even be subjected to capping policies. Thus, the design of current methods is driven by an end-to-end network philosophy that relies solely on a sender to a receiver communication.

The nature of managed networks calls for some properties that are currently missing in the design of current popular methods. These properties flip the underlying premises of popular adaptive streaming by regarding the network as a 'cooperative' element that works together with the server and the client to optimize video delivery. Specifically, the following principles need to exist in such solutions:

- The ERM, rather than the client, is best suited to control the quality switching mechanism, as it has full visibility with respect to the availability of network resources.
- The notion of a session needs to be established within the MSO's network to enable a more efficient and consistent management of QoE to the user.
- ERM should take into account the **physical** structure of the network, specifically the QAM resources. This is in contrast to popular adaptive streaming methods that model the network as having a flat sender-to-receiver logical links.

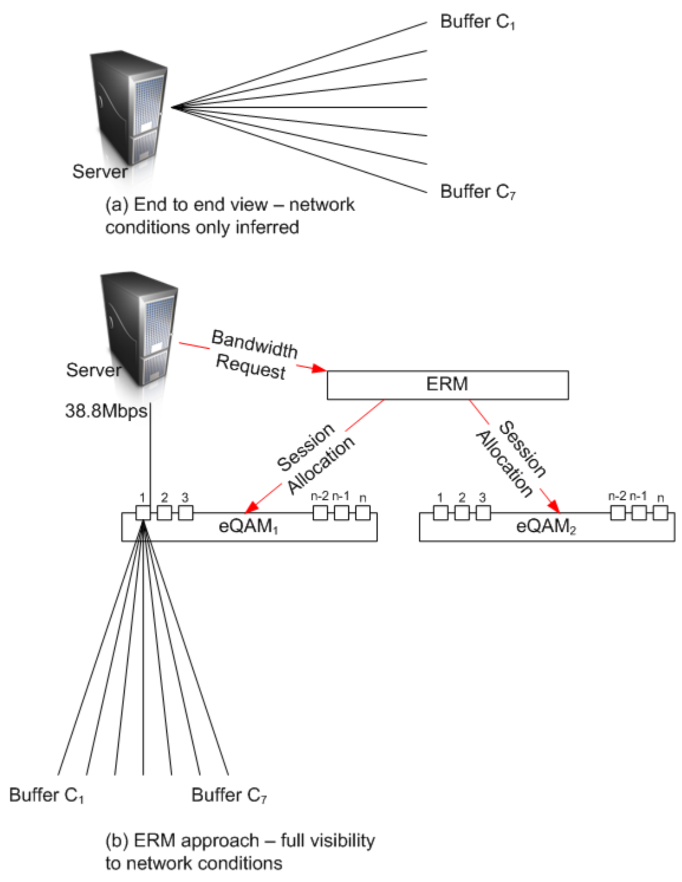


Figure 5: E2E vs. ERM-driven approach

In addition, some of the following are additional desired objectives:

- Reducing upstream control traffic from the client since the ERM takes on the role of switching decisions.
- Consequently, fragment size could be kept to minimum in order to provide better granularity in shaping the traffic.
- In some cases, content could be delivered from network edges relinquishing the need for fragment caching.
- Client control traffic may be used to provide valuable feedback on client perception of the network connection or its ability to receive traffic in a given rate.
- Supporting existing protocols to leverage the massive install base of adaptive streaming clients is highly desirable.

Switched Variable Bit Rate

A special case of adaptive streaming has been suggested previously by vendors in the industry for management of MPEG-2 TS services for STB devices. This method enables the delivery of video over an ERM eQAM architecture meeting many of the objectives that this paper enumerated previously. Figure 7 demonstrates how this method can be further expanded to support IP video over common HTTP protocols.

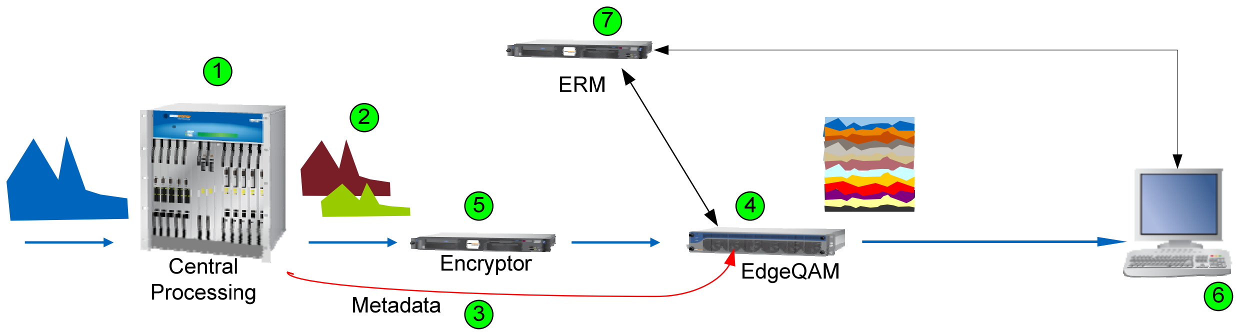


Figure 6: Switched Variable Bit Rate Support for MPEG-2/IP Video

Figure 7 illustrates a schematic view of how a network-centric adaptive streaming architecture would work. Content is being processed once per on-demand asset or per a broadcast channel at a central location (1). The processing includes the generation of multiple representations of the stream in several compression levels (2).

Additionally, metadata is generated to provide the *dictionary* aspect of the resultant content (3). The content and the metadata are then distributed to the network edge. The content may be cached at an edge device or delivered directly as a broadcast channel towards the edge QAM (4). It may be encrypted in the process (5) so long as the metadata remains in the clear. Further, a client device (6) establishes a VOD or SDV session with the network based on common VOD/SDV protocols. The session setup includes communication with the ERM (7) that provisions a session with the edge QAM with some bandwidth bounds. Based on the total number of incoming streams at a certain QAM channel and the level of picture complexity, the edge device will utilize dynamic quality switching to ensure the total outgoing bandwidth doesn't exceed the channel capacity.

This method adheres to the principles of adaptive streaming in the following ways:

- There is a composition of a video syntax that allows quality switching;
- There is a monitoring mechanism to detect congestions at network links;
- There is a decision function that triggers quality switching for a given content/stream; and
- There is a protocol (potentially internally inside the edge QAM) to switch from one representation to another.

Additionally, this method also includes many of the advantages of managed networks as discussed above:

- A network-centric admission control function;
- Network resources that are physically managed on the QAM level;
- The network (ERM and edge device) acts as the decision point for quality switching;

- There is no need for fragmentation of any size as switching can be achieved on a frame-level accuracy, which translates to:
    - A better user experience enabling switching across adequate points from a video encoding perspective;
    - Traffic that can be shaped very accurately to a target average bitrate, as opposed to pretty crude VBR changes used in common adaptive streaming methods. This accurate shaping contributes to more efficient management of the resources and potentially yields a higher QAM throughput; and
    - There is no upstream traffic (save for session setup/teardown and trick-play control).

Supporting the Installed Base

As we demonstrated, switched VBR is not limited to MPEG-2 TS or STBs and can be applied independently of the delivery channel or the transport protocols. Specifically, it can be utilized to provide IP video traffic compliant with existing Microsoft, Adobe or Apple implementations while still preserving the characteristics of network-centric processing.

Three additional goals still need to be met in order to support today's installed base of IP video players:

- Fragments must be generated in real-time by the edge device. The fragmentation would merely be a cropping of the continuous stream along the time points as described in the content manifest file;
- The switching protocol between the client and the server must be allowed. However, the server will not interpret this protocol as directives to switch quality but rather as client feedback as part of the monitoring process; and
- The content dictionary must exist and represent a *simplified* view of the content with discrete quality levels and fragment sizes.

Summary

Adaptive streaming is an effective method to bring together the robustness and efficiency of IP traffic with the special characteristics of video. Cable operators should not ignore the massive installed base of IP video players provided by Microsoft, Adobe and Apple if they hope to reach end consumers *Everywhere.* Consequently, the advantages provided by common adaptive streaming methods are a viable way to implement IP delivery with a better QoE for users while respecting network conditions. However, operators should also take hold of their competitive advantage in managing the delivery network and leveraging the investment in building a robust IP/video network by ensuring that IP video strategy is in line with existing ERM architecture. The approach outlined in this paper offers ways in which MSOs can harmonize the streaming and network architectures in order to fully realize the advantages provided by these technologies

Acronym List

Content Delivery Network	(CDN)
Cable Modem Termination System	(CMTS)
Edge Resource Manager	(ERM)
High Speed Data	(HSD)
Hypertext Transfer Protocol	(HTTP)
IP Multimedia Subsystem	(IMS)
Multimedia Messaging Service	(MMS)
Over the Top	(OTT)
PacketCable Multimedia	(PCMM)
Quality of Experience	(QoE)
Quality of Serivce	(QoS)
Real Time Messaging Protocol	(RTMP)
Switched Digital Video	(SDV)
Session Initiation Protocol	(SIP)
Session Manager	(SM)
Transmission Control Protocol/Internet Protocol	(TCP/IP)
User Datagram Protocol	(UDP)
Video on Demand	(VOD)