UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

DESIGN AND IMPLEMENTATION OF AN ABR VIDEO STREAMING SIMULATION MODULE FOR NS-3. ANALYSIS AND COMPARISON OF ABR VIDEO STREAMING ALGORITHMS OVER VARIOUS MOBILE NETWORK SCENARIOS.

XINXIN LIU

JUNIO 2021

TRABAJO DE FIN DE MÁSTER

Título:	Diseño e implementación de un módulo de ABR video streaming para NS-3. Análisis y comparación de algoritmos de ABR video streaming sobre varios escenarios de redes móviles.
Título (inglés):	Design and implementation of an ABR video streaming simulation module for NS-3. Analysis and comparison of ABR video streaming algorithms over various mobile network scenarios.
Autor:	Xinxin Liu
Tutor:	Marcus Ihlar (Ericsson AB)
Ponente:	Carlos Mariano Lentisco Sanchez (ETSIT-UPM)
Departamento:	Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:	—
Vocal:	—
Secretario:	—
Suplente:	—

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos

TRABAJO FIN DE MÁSTER

DESIGN AND IMPLEMENTATION OF AN ABR VIDEO STREAMING SIMULATION MODULE FOR NS-3. ANALYSIS AND COMPARISON OF ABR VIDEO STREAMING ALGORITHMS OVER VARIOUS MOBILE NETWORK SCENARIOS.

Xinxin Liu

Junio 2021

Resumen

El streaming de vídeo con tasa de bits adaptativa se está convirtiendo en la técnica más utilizada para las plataformas de vídeo en línea. Con la pandemia mundial *COVID-19*, el streaming de vídeo se ha convertido en una de las principales fuentes de entretenimiento durante los confinamientos. De hecho, más de la mitad de la cuota de tráfico de la red se utiliza hoy en día para streaming de vídeo [6].

El objetivo de este Trabajo Fín de Máster es construir un framework en *ns-3*, implementado en *C++*, para probar algoritmos de adaptación de vídeo y comparar algunas implementaciones sobre diferentes escenarios de red. El primer paso es estudiar *ns-3*, familiarizarse con algunos módulos de *ns-3* y construir varios escenarios de red *LTE*. El segundo paso es construir un módulo que pueda simular servidores y clientes de vídeo *ABR*, estudiar algunos enfoques de los algoritmos de adaptación de la tasa de bits de vídeo e implementar dichos algoritmos, incluyendo soluciones basadas en el ancho de banda, en el buffer y algoritmos híbridos. Por último, podemos comparar y evaluar el rendimiento de diferentes algoritmos *ABR* en escenarios con condiciones variables con diferentes métricas objetivas de *QoE*.

//// Resultados

Este proyecto se ha llevado a cabo con la cátedra Ericsson-UPM en software y sistemas.

Palabras clave: DASH, ABR, ns-3, streaming de video por HTTP, simulación, QoE

Abstract

Adaptive bitrate video streaming is becoming the most used technique for online video platforms. With the *COVID-19* worldwide pandemic, video streaming has become one of the primary sources of entertainment during the shutdown. In fact, more than half of the network traffic share today is used by video streaming [6].

The objective of this Master's Thesis is to build a framework in *ns-3*, implemented in *C++*, for testing video adaptation algorithms and to compare some implementations over different network scenarios. The first step is to study *ns-3*, familiarize with some *ns-3* modules, and build various LTE network scenarios. The second step is to build a module that can simulate *ABR* video servers and clients, study some approaches of video bitrate adaptation algorithms and implement those algorithms, including throughput based, buffer based and hybrid solutions. Finally we can compare and evaluate the performance of different *ABR* algorithms on scenarios with varying conditions with different objective *QoE* metrics.

//// Resultados

This project has been carried out with the Ericsson-UPM scholarship in software and systems.

Keywords: DASH, ABR, ns-3, HTTP video streaming, simulation, QoE

Acknowledgements

Contents

List of Figures

Glossary

IP - Internet Protocol

ABR - Adaptive BitRate

HTTP - HyperText Transfer Protocol

CPU - Central Processing Unit

DASH - Dynamic Adaptive Streaming over HTTP

MPEG - Moving Picture Experts Group

ISO - International Organization for Standarization

IEC - International Electrotechnical Commision

MPD - Media Presentation Description

URL - Universal Resource Locators

QoE - Quality of Experience

HLS - HTTP Live Streaming

ns-3 - network simulator 3

LENA - LTE-EPC Network simulAtor

DRM - Digital Rights Management

IIS - Internet Information Services

MSS - Microsoft Smooth Streaming

HDS - HTTP Dynamic Streaming

OSMF - Open Source Media Framework

IETF - Internet Engineering Task Force

3GPP - 3rd Generation Partnership Project

CDN - Content Delivery Network

NAT - Network Address Translation

XML - eXtensible Markup Language

Chapter 1 Introduction

1.1 Context

There is no doubt about the importance of online video streaming. According to Sandvine [6], in 2020, 57% of the global internet traffic was used by video streaming. Moreover, one of the key predictions made by Cisco in 2018 [7] stated that by year 2022, video traffic will make up 82% of all *IP* traffic.

Consequently, many challenges arise. Due to the growth of the number and diversity of video capable connected devices and every time more available bandwidth and better quality contents, the client and the server need to adapt the video content to the network and the devices. The technique of taking account the varying network conditions and computing resources of the user device to choose the adequate quality level is denominated as *Adaptive BitRate (ABR)*. Adaptation may be performed monitoring different parameters such as estimated bandwidth, client's buffer level, CPU load or screen size.

The *Dynamic Adaptive Streaming over HTTP (DASH)* is one of the standards that implements adaptive bitrate video streaming and was developed by the *Moving Picture Experts Group (MPEG)* [13]. *MPEG-DASH* enables provisioning and delivering media using existing *HTTP*-delivery networks supports dynamic adaptation with seamless switching. By using *HTTP*, the player will not have firewall problems, it will have better scalability and the quality selection relays on the client and there is no need to have session at the server.

The *MPEG-DASH* standard was published in 2012 and revised in 2019 by the *International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC)* as *MPEG-DASH ISO/IEC 23009-1:2019* [11]. In addition, the *3rd Generation Partnership Project (3GPP)* define the use of *DASH* as the standard continuous delivering of multimedia content in mobile networks, specifically in 4G such as *LTE* and 5G networks.

DASH divides the media file into small chunks or segments. *MPEG-DASH* defines the *Media Presentation Description (MPD)*, which is an XML-structured manifest file that

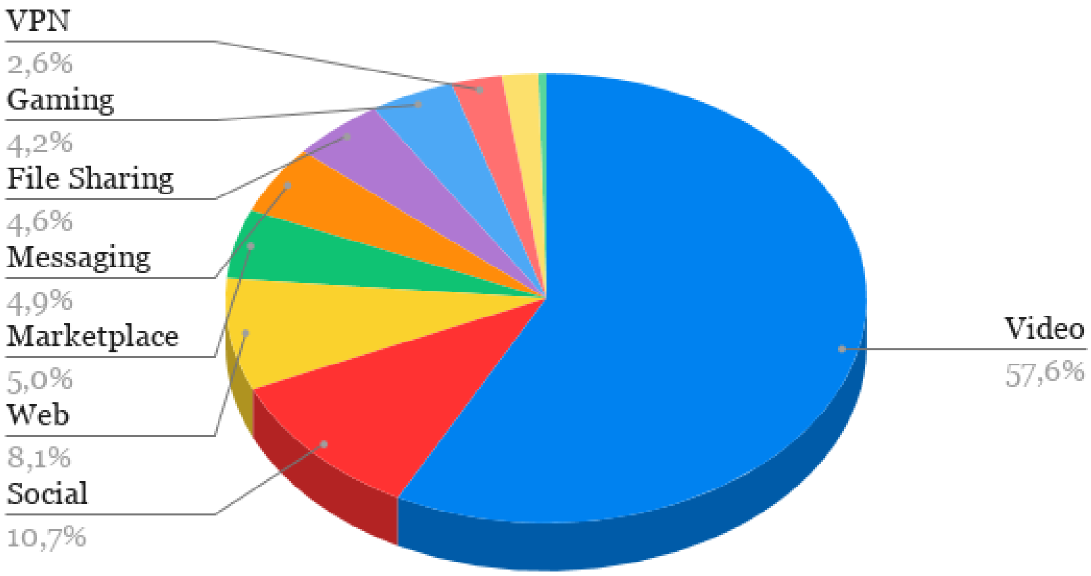

Figure 1.1: *Global application category total traffic share during COVID-19 lockdown. Source: Sandvine [6]*

contains the *Universal Resource Locators (URL)* of the segments. Different qualities are defined as representations, the *MPD* file contains information for each representation such as the codec, bandwidth, the resolution of the video or framerate.

However, the DASH Standard [11] only defines the data formats for the media reproduction and do not provide the adaptation algorithm. The *DASH Industry Forum* [8] provides an open source *MPEG-DASH* player implemented in *JavaScript* with different adaptation algorithms. Similarly, *hls.js* is an implementation of a *HTTP Live Streaming*[1] client.

The adaptation algorithms needs to be tested in different scenarios (real or simulated) and tweaked to provide the maximum perceived quality by the users. Also, there are algorithms that perform better in some specific scenarios and worse in others. The adaptation algorithm is the responsible of avoid problems that have a negative impact on the *Quality of Experience (QoE)*. Firstly, the algorithm can overestimate the bandwidth and it would cause a pause in the reproduction because all the segments in the buffer is emptied. The algorithm can also underestimate the bandwidth, the video player requests media segments with inferior quality than the quality at which the bandwidth available of the network can allow. Lastly, the algorithm should avoid constant bitrate switches result of bandwidth

[1]HTTP Live Streaming is a HTTP-based adaptive bitrate streaming protocol developed by Apple Inc. [3]

fluctuations, and provide a smooth and seamless video watching experience.

The *ns-3* simulator is an open-source and extensible discrete-event network simulator. The extensible nature of this tool allows us to develop a new module for *ns-3* mimicking the behaviour of *ABR* clients and servers. With this new module, *ns-3* will be able to simulate extreme network scenarios and test the performance of various adaptation algorithms.

1.2 Objectives

The objectives of this thesis is to build a framework for testing *ABR* adaptation algorithms, and implement some adaptation algorithms and compare them in various mobile network scenarios with different objective *QoE* metrics. In order to achieve the proposed objectives, the following steps will be proposed:

1. Study and understand *ns-3* and basic modules such as the core module, the internet module, applications module, *LENA* module among others. Build basic *LTE* scenarios tweak radio parameters, and output results.
2. Design a new module in ns-3 that simulates behaviours of *ABR* clients and servers. Study and implement existing adaptation algorithms.
3. Define and implement objective *QoE* metrics. Build new *LTE* scenarios and compare the performances of the implemented adaptation algorithms.

1.3 Structure of the thesis

Chapter 1. Presents the context, the motivations and the objectives of this thesis.

Chapter 2. The State of the Art. BBBBBBB

Chapter 3. dddd

Chapter 4. dddd

Chapter 5. dddd

Chapter 2 State of the art

In this chapter we ....

2.1 ABR Video Streaming

There are three ways of media delivery over *HTTP*. The first method is by **file download**, the media file is entirely stored in a local hard drive and played afterwards. The second method is called **progressive download**, in which the file is stored in a local hard drive but instead the download starts from the beginning and the media can be played when enough data are available. However, these two methods have disadvantages like waste of bandwidth, *DRM* issues and also requiring a reliable transmission. The last method is called **streaming**, contrary to the former two, the file is not stored locally, but played from the server, the client needs a data buffer to store the data that is being downloaded and when the session is closed the data are deleted.

Streaming media also comes with some challenges. There are a lot of network variability and a big heterogeneity in video capable devices. Therefore, to solve these shortcomings, *Adaptive bitrate streaming (ABR)* was created.

The basic idea of *Adaptive bitrate streaming* is to adapt the media content for the user by monitoring different parameters like estimated bandwidth, buffer level or *CPU load*, see Figure 2.1. There are many propietary adaptive streaming solutions:

- ***Apple HTTP Live Streaming (HLS)***: *HTTP Live Streaming HLS* is an implementation of an *ABR* protocol over *HTTP* developed by Apple [3] as part of the QuickTime software and the mobile operating system *iOS*. *HLS* supports live streaming and video on demand. *HLS* is proposed in 2009 as a standard to the *IETF* [12].
- ***Microsoft Smooth Streaming (MSS)***: *Smooth Streaming* is part of *Internet Information Services (IIS) Media Services* for delivering media over *HTTP* [14]. A prototype version of *Smooth Streaming* was used to deliver live and on-demand streaming

content from such events as the Summer Olympic Games in Beijing and the Democratic National Convention in Denver.

- ***Adobe HTTP Dynamic Streaming (HDS)***: *HTTP Dynamic Streaming* is the implementation of adaptive streaming by Adobe. *HDS* enables high-quality, network efficient HTTP streaming for media delivery that is tightly integrated with Adobe software [2]. The solution is based in using *Open Source Media Framework (OSMF)* and Adobe Flash Player.

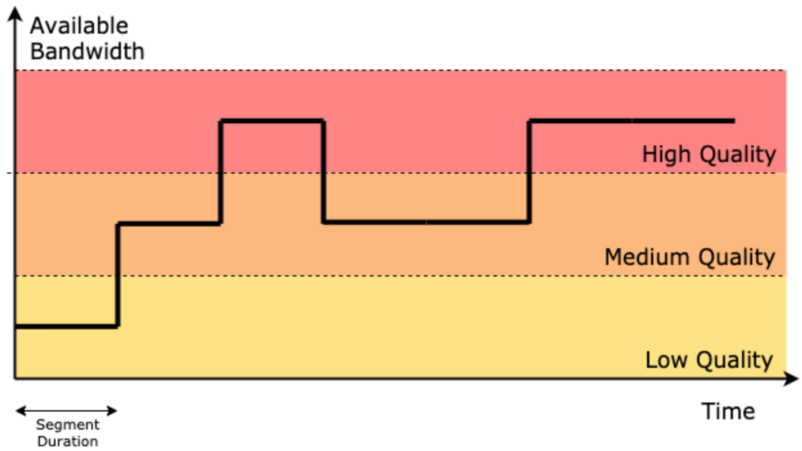

Figure 2.1: *Evolution of segment quality with time*

But there was no official standarization for adaptive video delivery over HTTP. For that reason, a new international stadard called *MPEG-DASH* was developed and published.

2.2 Dynamic Adaptive Streaming over HTTP

The *DASH* standard was created between the *Moving Picture Experts Group* from *ISO/IEC* and the *3GPP*. The development for *DASH* started in January 2009 and completed in March 2010. *MPEG-DASH* was published in April 2012 but has been revised in 2019 as *MPEG-DASH ISO/IEC 23009-1:2019* [11]. The *3rd Generation Partnership Project*

The objective of *DASH* was to create a unique standard that replaces the propietary solutions from Microsoft, Apple and Adobe. Also, it will offer the interoperability and the convergence necessary for the growth of big scale video streaming solutions. Microsoft, Apple, Netflix, Qualcomm, Ericsson and Samsung also took part of the development of the standard.

One of the biggest advantages of *DASH* is that the video streaming is over *HTTP* version 1.1 protocol (*HTTP/1.1*). The use of *HTTP* means that reusing existing internet infrastructure and media content distribution tecniques using *CDN (Content Delivery Networks)* can be done. Another convenience of using *DASH* is that due to using *HTTP* encapsulation, problems with passing through firewalls and the *Network Address Translation (NAT)* are not existent.

All the control of the media content delivery is located in the *DASH* client side. The standard does not define any web delivery mechanism nor the bitrate adaptation algorithm. What *DASH* does define in [11] are:

- ***The Media Presentation Description (MPD) File Format***: The *MPD* file uses the *eXtensible Markup Language (XML)* and contains the specifications of the media content and the *URL* of the segments in the *HTTP* video servers.
- **Segment format**: *DASH* defines the characteristics of the necessary codifications and the way that the media content is divided in small fragments called *segments*.

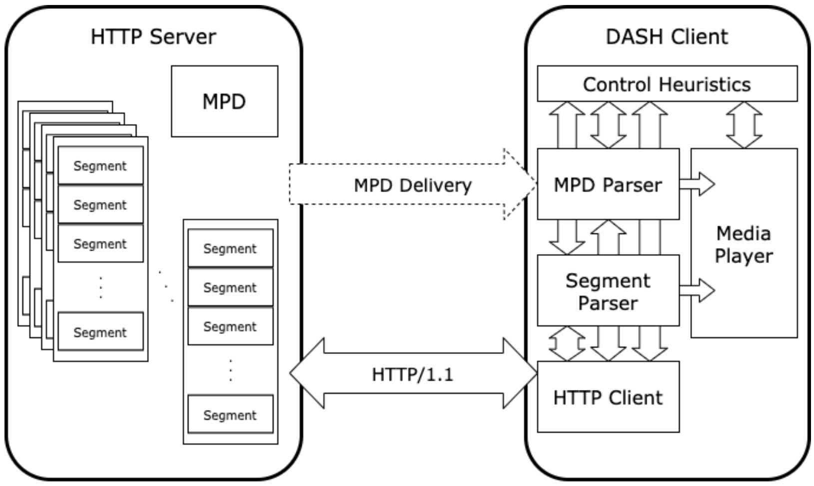

Figure 2.2: *DASH client-server architecture. Source: MPEG [17]*

The Figure 2.2 presents a simple *DASH* architecture. The video and audio content are processed and stored on an *HTTP* server. To access the content, the client sends *HTTP* requests to the server. But first, the client needs to download the *MPD* file, normally through *HTTP*. The client then does the parsing of the *MPD*, extract information such as the duration of a segment, media types or resolutions. Finally, the *DASH* client chooses the adequate quality and starts the streaming of the content using *HTTP GET* request to fetch the segments.

The *DASH* client stores the segments in a buffer and consumes the content. It continues to fetch new segments and by monitoring network variables it will decide which quality (higher or lower bitrate) to request next to avoid problems like buffer underflow and maintain at least a set number of segments in the buffer.

2.2.1 MPD

The *MPD* file is an *XML* document that describes the characteristics of the different media components that composes the media content (e.g. video, audio, subtitles).

The structure of the *MPD* is hierarchical. The media is devided in a sequence of **Periods** in time.

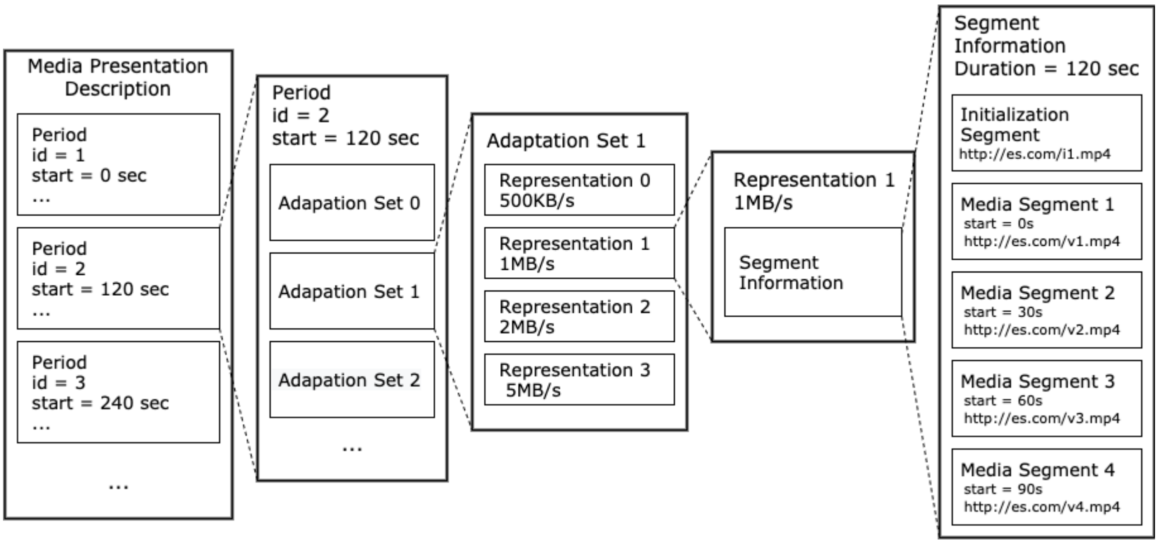

Figure 2.3: *The MPD hierarchical data model. Source: MPEG [17]*

2.3 Mobile Networks

2.4 Network Simulator 3

Chapter 3 | Conclusions And Future Work

Bibliography

[1] 3GPP. *Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH).* 3GPP TS 26.247 v16.4.1, 2020.

[2] Adobe. Live Streaming. https://www.adobe.com/es/products/hds-dynamic-streaming.html.

[3] Apple. HTTP Live Streaming. https://developer.apple.com/streaming.

[4] Benny Bing. *Next-generation video coding and streaming.* Wiley, 1st edition, 2015.

[5] Julián Cabrera. *Apuntes de Sistemas y Servicios de Multimedia (MUIT).* Universidad Politécnica de Madrid, 2020.

[6] Lyn Cantor. The global internet phenomena report covid-19 spotlight. Technical report, Sandvine, 2020.

[7] Cisco. Cisco predicts more ip traffic in the next five years than in the history of the internet. https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1955935, 11 2018.

[8] DASH-IF. DASH Industry Forum. https://dashif.org/.

[9] DASH-IF. dash.js. https://github.com/Dash-Industry-Forum/dash.js.

[10] Universidad Politécnica de Valencia. Vídeo adaptativo a través de MPEG-DASH. https://www.comm.upv.es/es/dash/.

[11] International Organization for Standardization. *Information technology — Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats.* International Organization for Standardization, ISO/IEC 23009-1:2019(E) edition, 2019.

[12] IETF. RFC 8216 - HTTP Live Streaming - IETF Tools. https://tools.ietf.org/html/rfc8216, August 2017.

[13] ISO. MEPG-DASH. http://www.iso.org/iso/home/standards.htm.

[14] Microsoft. Silverlight. https://www.microsoft.com/silverlight/smoothstreaming/.

[15] ns 3. A Discrete-Event Network Simulator. https://www.nsnam.org/.

[16] Miguel Ángel Aguayo Ortuño. Contribución a los mecanismos de adaptación dinámica para servicios de distribución multimedia sobre redes móviles. December 2020.

[17] Iraj Sodagar. White paper on MPEG-DASH Standard. MPEG-DASH: The Standard for Multimedia Streaming Over Internet. Technical report, ISO/IEC, 2012.

Chapter A | Impact

A.1 Social Impact

A.2 Economic Impact

A.3 Ambiental Impact

A.4 Ethic Impact

Chapter B | Budget