Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación


# EVALUACIÓN DE MÉTODOS DE OPTIMIZACIÓN DE TRÁFICO EN LA RED Y DESARROLLO DE ESCENARIOS EN REDES DE DISTRIBUCIÓN DE CONTENIDOS

## TRABAJO FIN DE MÁSTER

**Miguel Moisés Révolo Starke**

2015

Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación

**Máster Universitario en**
**Ingeniería de Redes y Servicios Telemáticos**

**TRABAJO FIN DE MÁSTER**

# EVALUACIÓN DE MÉTODOS DE OPTIMIZACIÓN DE TRÁFICO EN LA RED Y DESARROLLO DE ESCENARIOS EN REDES DE DISTRIBUCIÓN DE CONTENIDOS

Autor
**Miguel Moisés Révolo Starke**

Directora
**Encarnación Pastor Martín**

Departamento de Ingeniería de Sistemas Telemáticos

2015

## Resumen

El objetivo del proyecto de fin de máster es estudiar la distribución de vídeo actual, parte de su problemática y proponer una modificación a una aplicación existente que nos permita extender sus capacidades actuales y demostrar la importancia de la implementación de la inteligencia en las aplicaciones de lado de cliente. La tendencia actual señala que habrá un crecimiento importante en el consumo de vídeo en la Internet en los años venideros que demandará la ampliación o el uso eficiente de la capacidad de la red. Este incremento del consumo llevó a las distintas empresas a desarrollar tecnologías de distribución como lo es el streaming adaptativo que es la forma actual de transmisión de vídeo. También se revisó la forma en la que las redes de distribución de contenido (Akamai o Limelight) y los proveedores de contenidos (Netflix, Youtube, Hulu) operan. Se ha analizado el streaming adaptativo DASH para la visualización del contenido multimedia y se implementó una modificación sobre el comportamiento de una aplicación llamada dash.js para hacer que haga uso de varias fuentes de vídeo (que representarán dos CDNs posibles). Los cambios implementados ha sido puestos a prueba para ello se ha propuesto un escenario de red desarrollado en VNX. El escenario de red consistió en una red con dos fuentes de vídeo y un servidor web donde se cargaron máquinas virtuales con los vídeos y el cliente modificado. Adicionalmente se utilizó un shaper que degrada las condiciones de red entre el cliente y el primer servidor lo que fomenta al usuario a realizar el cambio al otro servidor de manera manual. El cambio de servidor no reinicio el video sino que continuó de manera transparente. Como resultados se han obtenido medidas de las trazas de depuración del modo de desarrollo web que nos indican como las calidades van cambiando de acuerdo al ancho de banda y que se podría obtener mejores resultado si se realizará un cambio de fuente. Lo estudiado y los resultados obtenidos dan luces de como a través de la utilización de métodos y herramientas de streaming adaptivo es posible mejorar la calidad de la los contenidos de audio y video.

## Abstract

The purpose of this work is to study video distribution, the problems associated with it and to propose a modification to an existing application. This modification will allow us to extend it capabilities and show the importance of client side intelligence. Recent surveys and forecasts indicate that there will be an important growth in the video consumption in the Internet in the next years. This growth will demand companies to increase network capacity or to use existing infrastructure more efficiently. As part of increasing efficiency adaptive streaming technology has been developed. Also the way content delivery networks (like Akamai and Limelight) and content providers (Netflix, Hulu and Youtube) operate have studied and investigated. Dynamic Adaptive Streaming over HTTP has been studied and an application called dash.js has been modified. The modification consisted in giving the user to make a manual change of the source of the video (which will represent two possible CDNs). The modified application has been deployed in a web server in an emulation scenario using the VNX tool. The scenario mainly consisted of two servers with the video files and a server with the modified client. Moreover a bandwidth shaper was used to degrade the throughput between the client and the first video source giving bad user experience. This bad conditions makes the user change from video source from video source 1 to 2 and improve its user experience (because server 2 has optimal network conditions). Results from this work are logs of the web browser console that show the changes in quality according to network conditions and show the improvements that can be made by choosing a good video source. This work shows how adaptive streaming can be improve by client intelligence and in doing so improve user experience with respect to video streaming.

# Índice General

# Índice de Figuras

## Índice de Tablas

## Lista de Acrónimos

**BIFS** Binary format for scene

**BML** Broadcasting markup language

**CDN** content delivery network

**CNAME** Canonical Name record

**CSS** Cascading Style Sheets

**DASH** Dynamic Adaptive Streaming over HTTP

**DLNA** Digital Living Network Alliance

**DNS** Domain Name System

**DOM** Document Object Model

**DRM** Digital Rights Management

**DVB** Digital Video Broadcasting

**FLUTE** File Delivery over Unidirectional Transport

**HTTP** Hypertext Transfer Protocol

**IGMP** Inter Group Management protocol

**IP** Internet Protocol

**ISP** Internet Service Provider

**ITU** International Telecommunication Union

**MPD** Media Presentation Description

**PIFF** Protected Interoperable File Format

**PoP** Point of Presence

**RTP** Real time Transport Protocol

**RTSP** Real Time Streaming Protocol

**SADS** Service and Application Discovery and Selection

**SAP** Stream Access Point

**SCP** Service Content Protection

**SCTP** Stream Control Transmission Protocol

**SST** Smooth Streaming Transport

**SVC** scalabe Vector Coding

**SVG** Scalable Vector Graphics

**TCP** Transmission Control Protocol

**UDP** User Datagram Protocol

**URI** Uniform Resource Identifier

**URL** Uniform Resource Locator

# 1 Introducción

## 1.1 Presentación

En los años recientes la Internet y las redes celulares han tenido un crecimiento importante. Este crecimiento ha generado un aumento del tráfico en la red. A su vez la masificación de los smartphones y tablets han contribuido con el aumento de datos en las redes móviles. Esto ha llevado a las empresas a desarrollar nuevas tecnologías como el 4G y el LTE. Con la nueva capacidad de la red se crea la posibilidad de ver vídeos a través de la Internet en especial en las redes móviles de manera masiva. Adicionalmente hay un crecimiento debido a empresas dedicadas solo a brindar contenido multimedia como Netflix, Hulu o Youtube. Las nuevas tecnologías como los smart-TV nos indican que el consumo de vídeo por Internet será la tendencia futura.

Una posible solución para afrontar este reto es incrementar la capacidad de red por medio de inversiones en infraestructura. Debido a que el despliegue de infraestructura es costoso se proponen soluciones de software y de mejora de los protocolos existentes. El crecimiento del sector hace que sea un área de investigación activa en la cual se indaga sobre todo el proceso de la distribución de vídeos, desde la codificación, los protocolos para su despliegue, la infraestructura de red necesaria, la protección de los derechos de autor (DRM), las recomendaciones a los usuarios, la tecnología para la reproducción, la monitorización de la calidad del vídeo y la satisfacción de los usuarios.

Este proyecto de fin de máster tiene por motivación estudiar los mecanismos de distribución de vídeo. También se busca implementar mejoras en los mecanismos de distribución existentes. En la actualidad gran parte del tráfico de Internet se debe a la transmisión de contenidos de vídeo y audio. Las decisiones de los proveedores de contenido como Netflix, Youtube y Hulu tienen un gran impacto en la red. Todo este tráfico sobrecarga las redes por lo que su estudio y compresión es importante para posibilitar un uso eficiente de soluciones de software y protocolos que incrementen la capacidad de la red para poder incorporar más usuarios en la misma red manteniendo la calidad de servicio que se complemente con una buena experiencia y satisfacción por parte del usuario.

## 1.2 Antecedentes

En los años 1990s la capacidad de procesamiento de los ordenadores no era suficiente para reproducir vídeo. Un problema a tener en cuenta en esos años era el hecho que la tecnología de red hacia uso de módems de 56k para la transmisión de información.

Entre los primeros reproductores y los más conocidos se encuentra el Windows Media Player y el Apple Quick Time que fueron lanzados en 1991.

En 1995 se realizó la primera transmisión en vivo de radio a través de Internet por ESPN SportZone. Netflix fue fundado en 1997 y en la actualidad tiene más de 50 millones de suscriptores en el mundo. Flash Player fue lanzado en el mismo año (1997) y sus principales funciones consistían en conseguir interactividad y media streaming. Al año siguiente en el 1998 se funda una de las principales redes de distribución de contenidos Akamai. [1]

En la década del 2000 había dos protocolos importantes para la distribución de vídeo: HTTP y UDP (además de muchos otros protocolos propietarios) pero sería HTTP el que ganaría la carrera al ser un protocolo de uso común y amplio. Es en la década del 2000 que las redes de distribución de contenidos (CDNs) se despliegan. La resolución de vídeo más común en el 2005 era de 320x240. El primer vídeo de Youtube fue subido a la red el 23 de abril de 2005. En su versión 7 Windows Media ya podía hacer uso de radios por internet y con la versión 10 ya podía acceder a tiendas online para la adquisición de música. Windows Media Player 11 y 12 ya tenían la capacidad de streaming.

En el 2010 sucedió un hito cuando el evento con mayor audiencia, la Copa Mundial de Futbol, fue visto en un porcentaje importante a través de la Web. En los Estados Unidos 45% de los televidentes vio los partidos fuera de casa y sin utilizar un televisor. Como consecuencia la página ESPN3.com tuvo más de 7 millones de usuarios y distribuyó millones de horas de contenido. Antes de ese gran evento se dieron algunos indicios de esa tendencia con los juegos de invierno en Vancouver 2010. De la población canadiense de 34 millones unos 4 millones vieron los eventos por Internet y se distribuyeron 6.2 Petabytes en dos semanas.

El nuevo perfil de los usuarios consiste en usuarios que quieren tener acceso a la mayor cantidad de contenido en cualquier momento. Esta tendencia se ha facilitado con el despliegue de redes de distribución de contenidos con tarifas más baratas. Páginas como Hulu se desplegaron aprovechando la situación. En el 2014 Hulu tenía más de 40 millones de usuarios en Estados Unidos y distribuía mil millones de horas de vídeo. El vídeo en Internet se puede clasificar en vídeos generados por usuarios amateurs y vídeos generados profesionalmente por estudios de cine o televisión. Los estudios suben sus vídeos con el propósito de promocionar sus productos cinematográficos y para vender directamente a los usuarios los vídeos. Las empresas de distribución de vídeo por Internet están proliferando ejemplos de esto son Netflix, Apple TV, AmazonVideo, Google Play y Waki.tv. Los índices de Cisco sugieren que se puede llegar a tener volúmenes de tráfico del orden de los exabytes y zetabytes.

Mejorar la calidad de la TV por Internet es un punto clave que influye en la experiencia que puede tener un usuario. De acuerdo a Cisco IBSG Youth Focus Group

Sessions se señala que de 7 de 8 factores de percepción del uso de la televisión de TV convencional y TV por cable ha arrojado que el único factor que aún permanece con menor puntuación es la calidad, la misma que influye en la percepción final de la experiencia en forma importante ya que la puntuación final se reduce a un 33% mientras que la TV convencional recibe una mejor puntuación de la calidad y consecuentemente la percepción final de la TV convencional es de 53% es decir mejor.

## 1.3 Definición del problema

Se quiere investigar y evaluar una aplicación (solución) que permita mejorar la calidad de la TV por Internet a través de una distribución de videos con una alta calidad. A través de la mejora los niveles audio-visuales que experimentan los usuarios de la TV por Internet serán mejores.

## 1.4 Relevancia del problema

El problema del aumento del tráfico de la red es muy importante debido a que por ejemplo durante la transmisión final de la Copa del Mundo 2014 Akamai llegó a tener un pico de tráfico de 6.62 Tbps, cabe mencionar que el mayor tráfico se cursó un par de días antes con 6.87 Tbps [2]. En el reporte de Akamai del estado de la Internet 2014 [3] se menciona que la velocidad promedio global es de 4.6 Mbps. El país con mayor velocidad promedio fue Corea del Sur con una velocidad de 24.6 Mbps. El país con mayor velocidad promedio de Europa fue Suiza con 14.9 Mbps. Cabe mencionar que España figura con una velocidad promedio de 8 Mbps que es casi el doble del promedio mundial. La velocidad promedio ha tenido un buen crecimiento a nivel global de 42% lo que demuestra el dinamismo del sector. Estos datos se muestran en la tabla 1.

| Ranking | País | Q2 2014 Velocidad Promedio (Mbps) | Cambio Anual (%) |
|---|---|---|---|
| | Global | 4.6 | 42 |
| 1 | Corea del Sur | 24.6 | 84 |
| 3 | Suiza | 14.9 | 35 |
| 14 | Estados Unidos | 11.4 | 39 |
| 34 | España | 8.0 | 36 |

Tabla 1: Velocidad Promedio

La velocidad pico promedio a nivel global fue de 25.4 Mbps. El país que reportó el mayor pico de tráfico fue Hong Kong con 74.9 Mbps. En Europa el país que tuvo el mayor pico fue Rumania con 63 Mbps. España está sobre el promedio global con 36.9

Mbps. Se puede apreciar que el crecimiento anual es un poco menor que la del crecimiento de la velocidad promedio. La información se muestra en la tabla 2.

| Ranking | País | Q2 2014 Velocidad Pico (Mbps) | Cambio Anual (%) |
|---|---|---|---|
| | Global | 25.4 | 34 |
| 1 | Hong Kong | 73.9 | 14 |
| 5 | Rumania | 63.0 | 33 |
| 17 | Estados Unidos | 45.3 | 30 |
| 39 | España | 36.9 | 15 |

**Tabla 2: Velocidad Pico**

Akamai en su informe tiene un índice que nos indica que porcentaje de las conexiones son mayores a 4 Mbps. Este índice es particularmente útil para determinar la capacidad de recibir vídeo en HD (720p y 1080) [4]. Estos datos se muestran en la tabla 3.

| Ranking | País | % con capacidad superior a 4 Mbps | Cambio Anual (%) |
|---|---|---|---|
| | Global | 59 | 18 |
| 1 | Corea del Sur | 95 | 11 |
| 3 | Suiza | 92 | 2.5 |
| 29 | España | 76 | 19 |
| 39 | Estados Unidos | 72 | 6.2 |

**Tabla 3: Cantidad de conexiones con capacidad mayor a 4 Mbps**

Dentro de las métricas consideradas por Akamai se menciona la cantidad de conexiones que son mayores a 10 Mbps. El país con más conexiones con capacidad mayor a 10 Mbps es Corea del Sur seguido de Suiza. España tiene un 20% de conexiones con capacidad mayor a 10 Mbps que es un poco menor que el promedio global. Estos datos se pueden apreciar en la tabla 4.

| Ranking | País | % con capacidad superior a 10 Mbps | Cambio Anual (%) |
|---|---|---|---|
| | Global | 23 | 65 |
| 1 | Corea del Sur | 78 | 72 |
| 3 | Suiza | 56 | 53 |
| 29 | Estados Unidos | 39 | 72 |
| 39 | España | 20 | 144 |

**Tabla 4: Cantidad de conexiones con capacidad mayor a 10 Mbps**

Un dato que llama la atención es que comentan el interés por la distribución de vídeo en calidad de ultra HD. Debido a dicho interés han creado una métrica para determinar qué tan preparados están los distintos países para distribuir vídeo en 4k. Los streams adaptativos en 4k por lo general hacen uso de 10 – 20 Mbps de ancho de banda. La métrica indica que países tienen mayor concentración de conexiones que hacen posible la distribución en ultra HD. Adicionalmente se menciona que el índice es nuevo y que la introducción de nuevos códecs como HEVC o VP9 pueden cambiar los requisitos para tener acceso a vídeos en 4k. De las conexiones de Akamai solo 12% tenían capacidad de 15Mpbs o superior. A continuación presentamos tablas del informe que nos dan una idea de la situación de conectividad a nivel mundial. La tabla 5 nos muestra algunos datos relevantes.

| Ranking | País | % con capacidad superior a 15 Mpbs | Cambio Anual (%) |
|---|---|---|---|
| | Global | 12 | 98 |
| 1 | Corea del Sur | 62 | 123 |
| 3 | Suiza | 33 | 108 |
| 29 | Estados Unidos | 19 | 122 |
| 39 | España | 9 | 198 |

**Tabla 5: Cantidad de conexiones con capacidad mayor a 15 Mbps**

En el reporte de Akamai hay una sección dedicada a las conexiones móviles. En esta se mencionan resultados muy interesantes que se presentan a continuación en la tabla 6. Se muestra los mejores resultados por región y el caso de España.

| Región | Velocidad móvil promedio (Mbps) | Pico de velocidad Mpbs | % de conexiones con más de 4Mbps |
|---|---|---|---|
| África | 2.3 (Egipto) | 13.2 (Marruecos) | 5.5% (Sud-Africa) |
| Asia Pacífico | 15.2 (Corea Sur) | 108.0 (Australia) | 76% (Corea Sur) |
| Europa | 8 (Eslovaquia) | 39.3 (Rusia) | 92% (Dinamarca) |
| América del Norte | 7 (Canadá) | 21.8 (Canadá) | 63% (Canadá) |
| América del Sur | 3.7 (Venezuela) | 16.8 (Venezuela) | 7% (Uruguay) |
| España | 5.1 | 28.9 | 47% |

Tabla 6: Conexiones Móviles

Adicionalmente se muestran resultados obtenidos por Ericsson. Debido a que Ericsson tiene presencia en 180 países y su base de clientes representa más de 1000 redes por lo que sus medidas del uso de voz y datos en los móviles son representativas del sector. En la gráfica se aprecia el importante crecimiento del sector móvil y el aumento del uso de datos con respecto a la voz. En la figura 1 se pueden ver sus resultados.

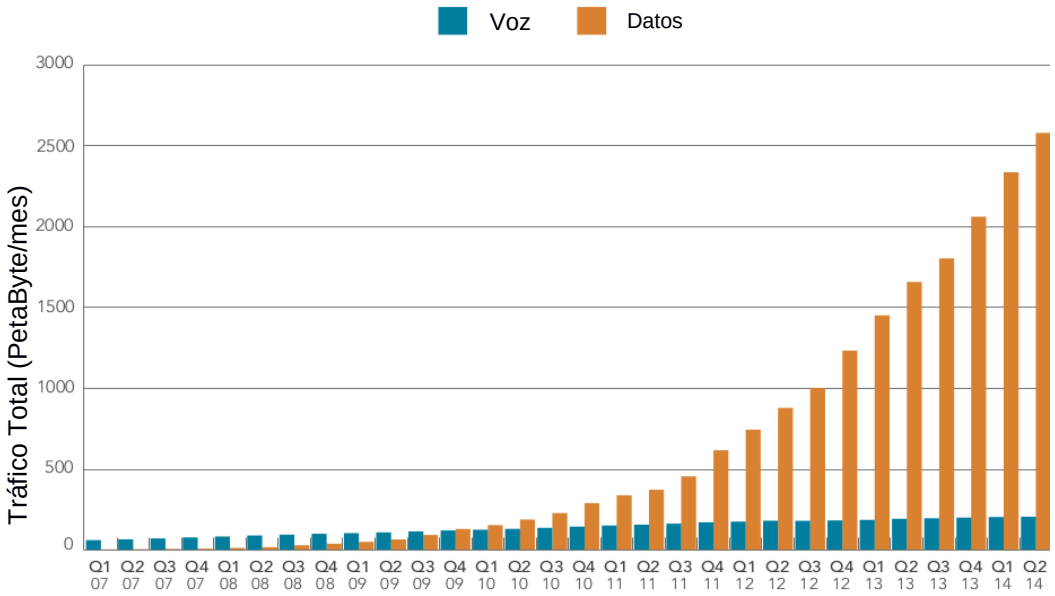


Figura 1. Crecimiento de datos móviles

Del reporte Índice Visual de Redes de Cisco [5] podemos mencionar datos muy interesantes sobre el sector móvil. El tráfico global llegó a 1.5 exabytes por mes a fines del 2013 esto representa un crecimiento importante considerando que a fines de 2012 el

tráfico móvil era de 820 petabytes. El vídeo móvil llegó a ser el 53% de todo el tráfico en el 2013. Durante el 2013 un dispositivo "smart" generó 29 veces más tráfico que uno convencional. En el 2013 las tablets se incrementaron 2.2 veces llegando a ser 98 millones y cada tablet generó 2.6 veces más tráfico que el smartphone promedio. También hubo 149 millones de laptops en las redes móviles en el 2013 y cada laptop generó 4.6 veces más tráfico que el smartphone promedio.

En el 2013 las conexiones de cuarta generación (4G) generaron más tráfico que las convencionales. A pesar de que las conexiones 4G representan solo el 2.9% de las conexiones móviles representan el 30% del tráfico móvil. El tráfico en Vodafone Europa creció un 35% durante el período 2012-2013. Para el año 2018 dos tercios de todo el tráfico será vídeo. El vídeo en dispositivos móviles se incrementará 14 veces entre el 2013 y el 2018. En la figura 2 que se muestran a continuación se puede apreciar las proyecciones de crecimiento de tráfico por región para el 2018.

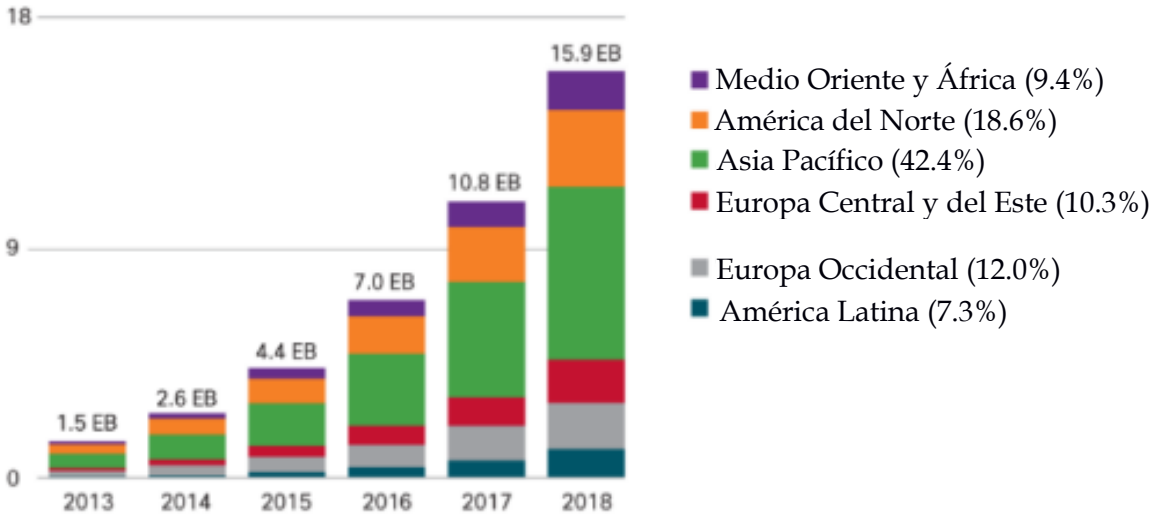


**Figura 2. Crecimiento de datos móviles por región**

En la figura 3 se muestra que el vídeo será una parte importante del tráfico total de la red. Esto se debe a que el vídeo tiene una mayor tasa de bits que otros contenidos móviles, el vídeo generará la mayor cantidad de tráfico a través del 2018. El vídeo crecerá con una tasa de crecimiento anual compuesto de 69% entre el 2013 y el 2018. De los 15.9 exabytes de tráfico mensual en el 2018, 11 exabytes serán de debido al uso de vídeo. El vídeo móvil representó más de la mitad del tráfico en el 2012 por lo que el vídeo móvil ya tiene un gran impacto en el presente.

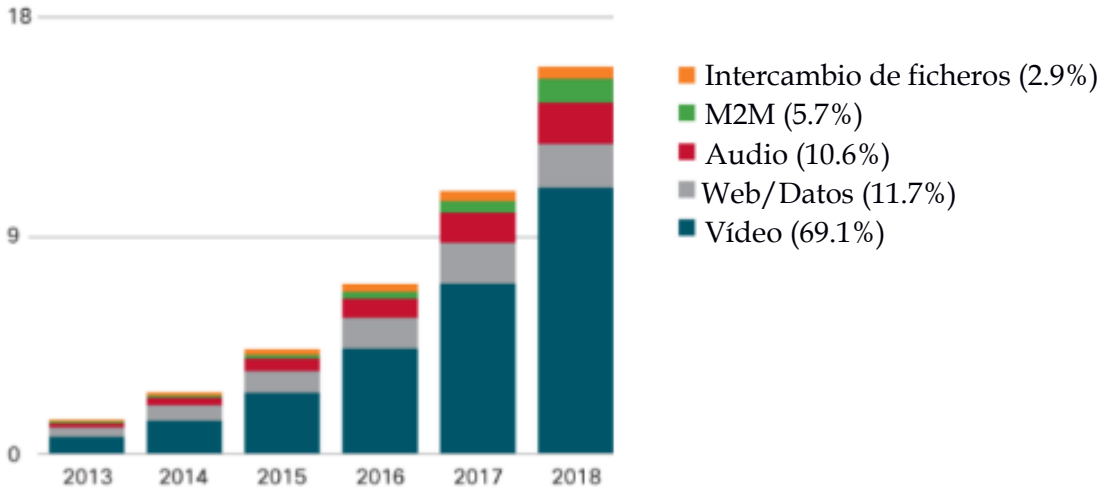


Figura 3. Crecimiento móvil por tipo de aplicación

De manera análoga los servicios de vídeo en la nube serán predominantes. Las aplicaciones en cloud serán el 90%. El tráfico de aplicaciones de tipo no cloud se reducirán al 10% del tráfico total. La figura 4 nos muestra las proyecciones que tendrá el tráfico en cloud.

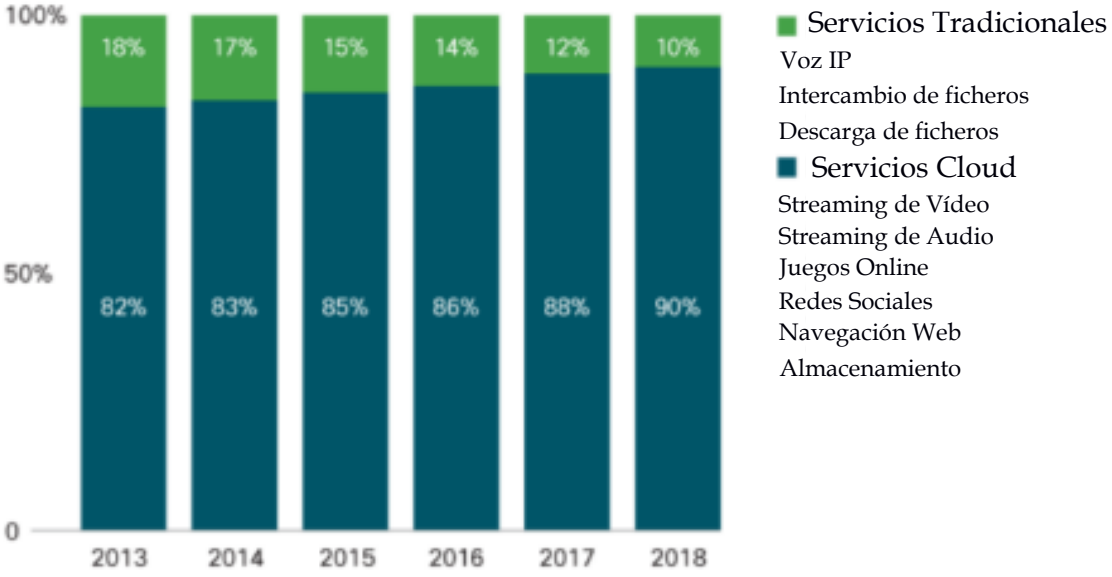


Figura 4. Crecimiento móvil por tipo de servicio

## 1.5 Propósito del estudio

El objetivo del estudio es implementar un escenario usando herramientas de virtualización para emular el proceso de distribución de vídeo. Este escenario tiene por finalidad ser una plataforma para implementar las mejoras que se proponen en la literatura sobre las mejoras en los criterios de distribución de vídeo streaming.

## 1.6 Limitaciones

Las limitaciones del presente estudio es el tiempo que se dispone para comprender los componentes de la aplicación dash.js. Otra limitación es el constante desarrollo del programa dash.js y de sus componentes. Cabe tener en cuenta que la documentación también está siendo revisada y actualizada constantemente.

## 1.7 Delimitaciones

El trabajo de esta delimitado a la versión 1.1.2 de la aplicación dash.js. El trabajo se enfocará en las modificaciones al dash.js para implementar un cambio manual de selección de la fuente del vídeo.

## 1.8 Resumen

El presente capítulo hemos analizado la importancia del sector del vídeo por Internet y como ha ido creciendo a lo largo de los años. Las estadísticas de Cisco y de Akamai indican que el consumo de vídeo es importante y que solo puede crecer en el futuro. En los últimos años ha aumentado vertiginosamente la cantidad de usuarios con capacidad de acceder a contenido de vídeos y eventos en la Internet por lo que es importante tenerlo en cuenta a la hora de diseñar las redes de telecomunicaciones. Al considerar el efecto del consumo de vídeo masivo se encuentran que las conexiones actuales pueden brindar un buen servicio para ciertas calidades medias de vídeos. Son pocos los países que pueden brindar servicios de vídeo de alta calidad y ultra alta calidad pero la tendencia es llegar a tener una capacidad de red que lo permita. La región de Asia-Pacífico es la que más capacidad tiene y la que más crecerá. Sin embargo todas las regiones tienen un buen crecimiento anual. Es en este contexto que la distribución de vídeo cobra importancia para que se pueda brindar un buen servicio. Si bien la mayor cantidad de tráfico de vídeo se generó en redes cableadas el sector móvil está creciendo rápidamente para satisfacer la demanda de los usuarios de acceder a contenidos desde sus dispositivos móviles.

# 2 Marco Teórico

En el marco teórico se presenta los conceptos claves de los métodos de optimización de tráfico en la red y desarrollo de escenarios en redes de distribución de contenido. Los componentes básicos de un sistema de distribución de vídeo son el servidor que ofrece el vídeo y cliente a través del cual el usuario solicita el acceso a los vídeos. Es importante mencionar que en el Internet actual no existen mecanismos incorporados para medir la calidad de servicio [32] [33].

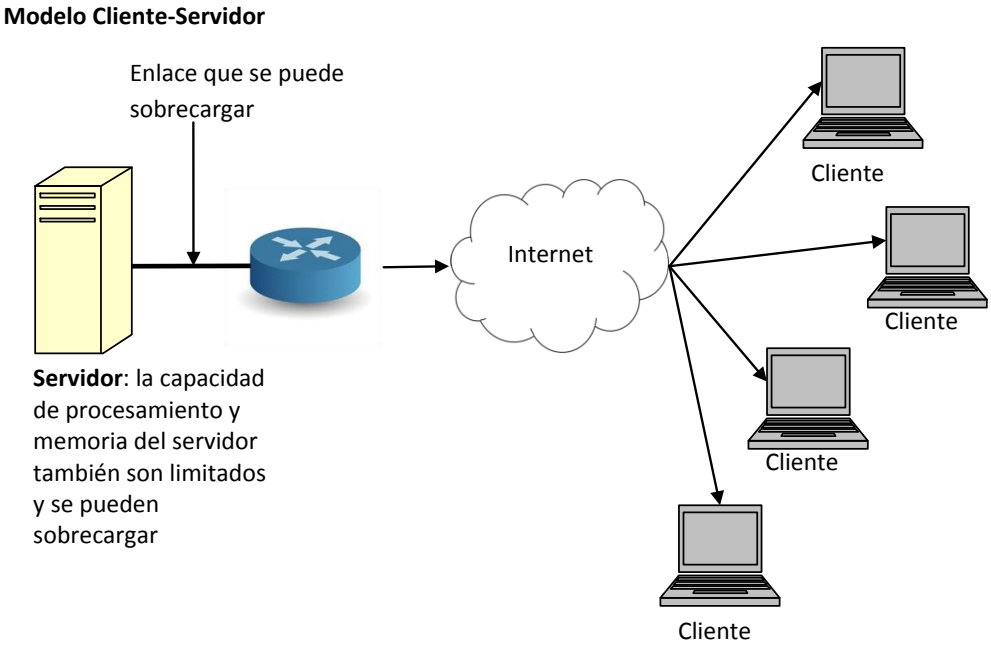


Figura 5. Modelo cliente servidor

Un componente importante del cliente es el buffer. El buffer almacena los datos descargados para luego reproducirlos. Con respecto al funcionamiento del buffer hay dos casos importantes si la velocidad de descarga es igual o mayor que la velocidad de reproducción entonces el usuario verá el contenido de manera fluida y sin interrupciones. En el caso de que la velocidad de descarga sea inferior a la de reproducción el usuario experimentará paradas y una reproducción discontinua [34].

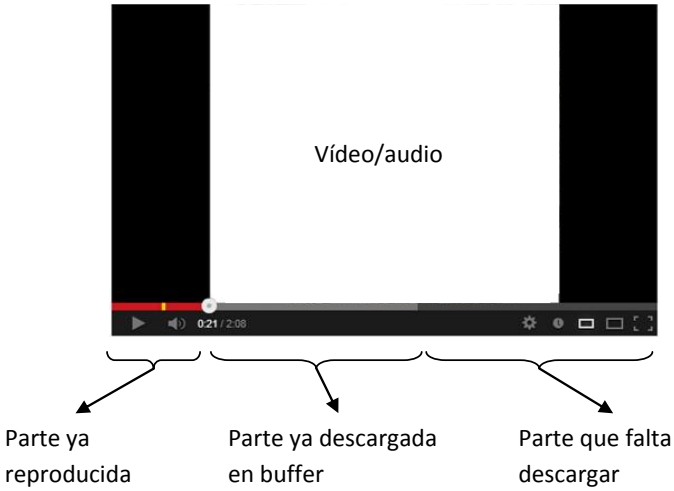


Figura 6. Reproductor de audio/vídeo y búffer

El protocolo HTTP es el protocolo de transferencia de hipertexto [29] [30] [31]. Su tarea fundamental es realizar peticiones HTTP GET o POST que suelen tener por lo general una respuesta HTTP con contenido en HTML. El navegador web como Internet Explorer, Firefox o Chrome (entre otros) realiza peticiones HTTP para obtener las páginas web que los usuarios desean obtener. El protocolo TCP [36] es el protocolo de transporte que se utiliza en conjunto con IP [35] para llevar las páginas web a los usuarios (pues estas páginas se segmentan y ensamblan al ser transportadas por la red). HTTP es el protocolo para solicitar páginas. TCP (junto con IP) es el protocolo que se encarga de hacer que lleguen a ti.

La popularidad de los protocolos HTTP y TCP se debe a su simplicidad. En el caso de una descarga continua de vídeo se puede empezar con la reproducción cuando se tiene suficiente información en el buffer. En el caso de descargas progresivas el usuario debe escoger la versión (la resolución) del vídeo a ver antes de empezar la descarga. Si el usuario no escoge la versión más apropiada para su ancho de banda el vídeo se cortará y tendrá que esperar a que el buffer se llene para poder seguir viendo. Para resolver el problema anterior se ha desarrollado el streaming adaptativo que permite seguir con la simplicidad del método de descarga progresiva. El streaming adaptativo es una combinación de la descarga progresiva y se basa en solicitudes pull de segmentos HTTP que el reproductor (cliente) solicita al servidor. Adicionalmente los segmentos HTTP son cortos lo que permite al cliente solo descargar lo que necesita y a su vez usar el "trickplay" de manera más eficiente. Otro detalle importante es que los segmentos MPEG4 están disponibles en muchos bitrates que se corresponden con las distintas resoluciones (calidad de imagen de vídeo). Un comportamiento destacable es el comportamiento del cliente que solicita el siguiente segmento HTTP al servidor basándose los recursos de la red, ancho de banda, estado de las conexiones TCP, capacidades de los dispositivos (como la resolución de la pantalla del usuario y

capacidad de CPU). El objetivo es proveer de la mejor experiencia de usuario a través de la mejor calidad de imagen posible, de un arranque rápido del vídeo y búsquedas más rápidas del vídeo [10]

Debido a que el streaming adaptativo hace uso de HTTP tiene una cobertura masiva al ser HTTP el protocolo por defecto de la Internet que todos los dispositivos manejan. Debido a que es un protocolo que hace uso de solicitudes pull puede ir a través de middlewares y NATs. Mantiene poca información del estado de la conexión lo que permite mayor escalabilidad (debido a que el servidor no tiene que guardar mucha información por conexión). Una ventaja adicional es que se puede utilizar las técnicas de cacheo HTTP para cualquier objeto HTTP y utilizar protocolos de tipo REST.

## 2.1 Media Streaming

La transmisión de contenidos en la red puede llevarse a cabo de distintas maneras. Los criterios fundamentales para determinar el método son: el tipo de contenido a transmitir y las condiciones de red. La tabla 7 muestra las condiciones de red y métodos respectivos de cada tipo de contenido.

| Tipo de contenido | Condiciones de Red | Método |
|---|---|---|
| Transferencia de Ficheros | Red con pérdidas de paquetes | Énfasis en distribuir el contenido con confianza por lo que se puede añadir redundancia para proteger los paquetes |
| Audio/Vídeo con requisitos de tiempo real | Red de alta velocidad | Baja latencia y jitter. Se requiere una transmisión eficiente |

**Tabla 7: Condiciones y métodos para la transmisión de contenidos**

### 2.1.1 Protocolo Tipo Push

En los protocolos tipo push el cliente y el servidor establecen una conexión. El servidor envía paquetes al cliente hasta que este corta la conexión o manda un mensaje que interrumpe o detiene la sesión. En sesiones tipo push el servidor está constantemente escuchando a las solicitudes del cliente (que son comandos de administración de la sesión). Uno de los protocolos más utilizados para conexiones tipo push es el protocolo RTSP (Real-time Streaming Protocol) [38] que administra la sesión del cliente y el servidor. Para la transmisión de datos se suele utilizar el protocolo RTP (Real-time Transport Protocol) que hace uso del protocolo UDP [39] sin tener ningún

tipo de control de velocidad pues esta se administrará al nivel de aplicación. Esto hace al protocolo RTP [40] una buena opción para transmitir información que requiera poca latencia y puede enviarse con el "mejor esfuerzo" (best effort). En sesiones convencionales de streaming push, el servidor envía la información a la velocidad de consumo del cliente. También existen propuestas de mejorar la distribución de datos haciendo un uso adicional de las técnicas pull [41]. En circunstancias normales ese comportamiento garantiza que la reproducción sea continua al mantener el buffer del cliente en un estado estable (con suficiente información para reproducir continuamente). Un beneficio de este método es el uso eficiente de la red porque el cliente por lo general no puede consumir más recursos que la velocidad del vídeo (o audio que este consumiendo) lo que evita sobrecargar a la red. Un posible problema se da en el caso que los paquetes se pierden lo que puede resultar en que el buffer no tenga información y el vídeo se detenga. Para prevenir que el buffer se vacíe se utiliza un cambio dinámico a menores velocidades de transmisión de datos (vídeo con menor calidad) así se contrarresta el efecto de la red (pero se mantiene la continuidad de la reproducción). La reducción de la velocidad se debe hacer progresivamente para que el usuario no note un cambio muy brusco en la calidad del contenido que está consumiendo. La calidad del contenido se puede mejorar de manera acorde a las condiciones de la red. Para conseguir este comportamiento se deben monitorizar las condiciones de la red. La monitorización de la red por lo general se realiza en el cliente que también calcula métricas de red (como el RTT, pérdida de paquetes, tiempo de retardos y diferencia de los tiempos de los retardos). El cliente puede usar esta información de manera directa para tomar decisiones respecto al momento de cambiar a una velocidad distinta. El cliente también puede enviar los resultados de las medidas al servidor (a través del protocolo RTP) para que este tome la decisión de la calidad del contenido [10].

### 2.1.2 Protocolo Tipo Pull

En protocolos tipo pull para streaming el cliente es una entidad activa que solicita contenido. La respuesta del servidor dependerá de las solicitudes del cliente adicionalmente dependerá si el servidor está disponible o bloqueado al cliente. Además es una forma muy robusta para solicitar e enviar vídeos aun cuando no se realicen medidas [42]. La tasa de bit a la cual el cliente recibe el flujo de información dependerá del cliente mismo (que solicita los paquetes) y las condiciones de red (que permite que los paquetes lleguen y brindan los parámetros de decisión al cliente). El principal protocolo de descarga en la Internet es el protocolo HTTP por lo que es la opción para la distribución de contenido tipo pull.

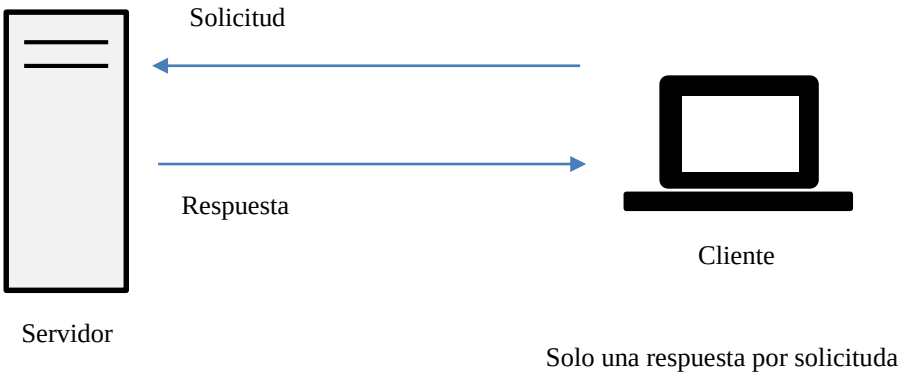


**Figura 7. Protocolo tipo pull**

La descarga progresiva es uno de los métodos más utilizados en streaming tipo pull. En una descarga progresiva el cliente envía una solicitud HTTP al servidor y empieza a solicitar el contenido del servidor tan rápido como sea posible. Una vez que el cliente llena el mínimo de buffer requerido, se empieza con la reproducción. El cliente continuará descargando segmentos del servidor como un subproceso durante la reproducción, mientras que la velocidad de descarga sea mayor que la velocidad de reproducción se puede ver el contenido de manera fluida y sin cortes. Sin embargo, si las condiciones de red se deteriorasen entonces se podría dar el caso que la tasa de descarga llegue a ser menor que la de reproducción lo que llevará a una interrupción de la reproducción en el cliente. Se utilizan métodos similares a los del tipo push con respecto al cambio de velocidades dependiendo de las condiciones de red. La adaptación de la tasa de bits se basa en dividir el contenido original en muchos segmentos de corta duración llamados fragmentos. Los fragmentos estarán codificados a distintas calidades (tasa de bits) y podrán ser decodificados de manera independiente. Cuando un cliente reproduce los fragmentos se puede reconstruir de manera adecuada el flujo del contenido original. Durante la reproducción el cliente pide fragmentos de acuerdo al ancho de banda disponible, de esta forma el cliente se adapta a la situación de la red. Las estructuras de los fragmentos podrán variar en las diferentes implementaciones pero el principio de funcionamiento es el mismo. Cuando el audio y el vídeo no están entrelazados (uso del 'Interleaving' para ayudar en la corrección de errores) cada segmento de audio consiste de muestra de audio en el rango de los milisegundos y cada segmento se puede decodificar independientemente. De esa forma se puede agregar los fragmentos de audio correspondientes a los enviados en vídeo. En el caso del vídeo puede ser que los fragmentos no sean independientes por los distintos métodos de predicción temporal y espacial que se usan en la codificación de los vídeos. Por esta razón la división del vídeo en fragmentos se realiza en un grupo de imágenes (GOP = group of pictures) [44]. Un GOP es una secuencia que empieza en una trama tipo I que puede decodificarse de manera independiente seguida de tramas calculadas a

partir de otras tramas. Si las tramas calculadas solo dependen de las tramas del mismo grupo se le considera un GOP cerrado y las tramas dependen de otros GOP se le considera como un GOP abierto. Como los GOP cerrado son autosuficientes e independientes su reproducción es directa. Si el paquete de las solicitudes y respuestas fuera más grande que los fragmentos se pueden enviar varios segmentos en la misma solicitud.

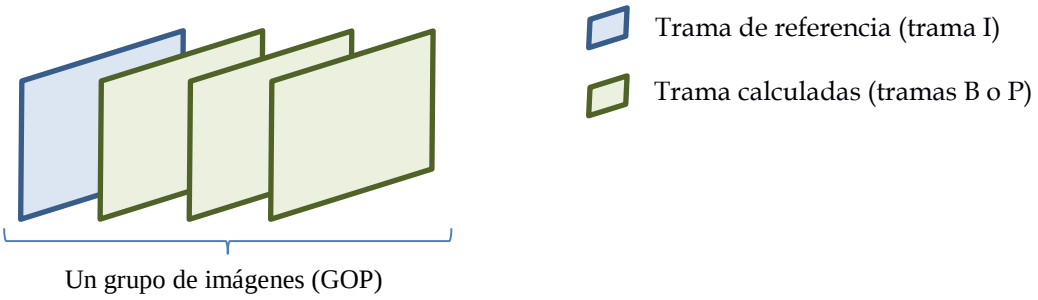


**Figura 8. Grupo de imágenes**

### 2.1.3 Microsoft Smooth Streaming

Basado en PIFT (Protected Interoperable File Format) [46], que es una extensión de la especificación MPEG4 (mp4) [43]. En Smooth Streaming todos los fragmentos con la misma tasa de bit se envían en el mismo fichero mp4. En consecuencia cada velocidad tiene un fichero distinto. Los fragmentos son usualmente de dos segundos de duración y contienen un solo GOP.

Un fichero mp4 consiste de información almacenada de forma estructurada. El elemento más básico de esta estructura es un contenedor que puede tener audio o vídeo. Cada contenedor tiene un identificador de cuatro letras. El fichero comienza con un contenedor que describe la versión para la cual es compatible. Seguidamente hay otro contenedor con información de los contenidos multimedia del fichero. En mp4 el contenedor mdat tiene el audio y el vídeo. En un mp4 segmentado el contenedor mdat tiene un contenedor moof que contiene la información específica de ese segmento de mp4. El contenedor moof puede tener información de señalización como la secuencia del fragmento, el número de muestras dentro del fragmento y la duración de cada muestra. Para que el cliente solicite un fragmento con una tasa específica primero necesita saber que fragmentos están disponibles en el servidor. Esta información se comunica a través de un fichero de manifiesto al inicio de la sesión entre el cliente y el reproductor. Una vez que el cliente tiene el fichero de manifiesto usa la información contenido en él para generar las solicitudes HTTP para pedir los fragmentos de vídeo del servidor. Cada fragmento se descarga solo con un par solicitud-respuesta HTTP. La

solicitud HTTP tiene dos parámetros: el bitrate y el tiempo (ubicación temporal) del fragmento. El servidor debe tener clasificados los segmentos y las calidades para esto el servidor tiene su propio fichero de manifiesto (server-side manifest) que realiza esta clasificación. El fichero mp4 tiene un contenedor mfra que contiene el tiempo del fragmento. Una vez que el servidor ubica el fragmento lo envía los contenedores moof y mdat que forman el stream [57]. Cabe mencionar que solo existe un fichero de vídeo y que los segmentos se crean de manera virtual lo que ofrece una buena capacidad de la administración del almacenamiento. [45]

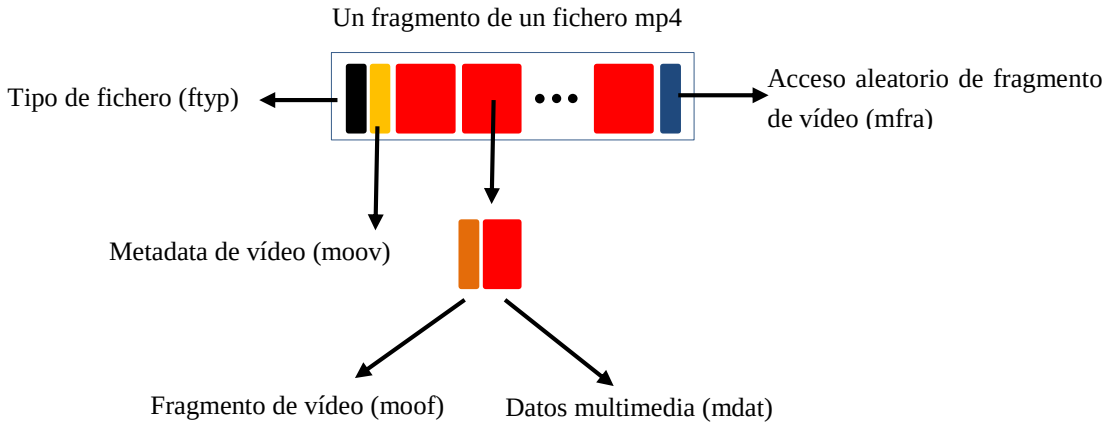


Figura 9. Fragmento de Microsoft Smooth Streaming

### 2.1.4 Apple HTTP Live Streaming

Apple tiene un enfoque distinto para el manejo de los fragmentos. La forma de almacenar los fragmentos en el sistema de Apple se basa en el ISO/IEC 13818-1 MPEG2. A diferencia del método de Smooth Streaming cada segmento se almacena en un fichero de transporte diferente. Un proceso de segmentación de contenidos divide el fichero de transporte en fragmentos de 10 segundos de duración. Si los fragmentos duran más entonces se tendrán menos fragmentos. Fragmentos más largos permiten una mejor compresión porque tienen más redundancia temporal. Sin embargo también se reduce la granularidad (la cantidad de niveles) posible sobre la cual tomar decisiones. Por lo que se generarían cambios muy abruptos de la calidad cuando se produzcan variaciones en las condiciones de red. De manera similar al Microsoft Smooth Streaming se tiene un fichero de manifiesto que lleva un registro de todos los segmentos disponibles al cliente. El formato del fichero de manifiesto es una extensión de una lista de reproducción mp3. Cada fichero empieza con una etiqueta EXTM3U que le distingue de una lista normal mp3. [10]

El archivo de manifiesto tiene los URI de los segmentos que son precedidos por la etiqueta EXT-X-STREAM-INF. Esta etiqueta tiene un atributo llamado Ancho de banda que indica los ficheros correspondientes para cada ancho de banda (calidad). En dichos

ficheros (fichero de mapeo de segmentos) ya se tiene la ubicación de los segmentos. En los ficheros de mapeo de segmentos se tiene la etiqueta EXT-X-MEDIA-SEQUENCE que lista el número de secuencia del primer segmento. Cada segmento tiene su propio número de secuencia. Los URI para cada segmento están marcados con una etiqueta llamada EXTINF. Esta etiqueta tiene un atributo que indica la duración del segmento. Para streams de duración fija la cadena de caracteres X-ENDLIST indica el final de la lista de reproducción. Esta cadena no existe para streams en vivo que no tienen una duración predeterminada. En el caso de streams de transmisiones en vivo el usuario tiene que descargarse periódicamente el manifiesto. El período de recarga del manifiesto depende si el fichero de manifiesto ha cambiado desde la última vez que se solicitó. Si el fichero de manifiesto ha cambiado el período es la duración del último segmento en la lista de reproducción. Si el manifiesto no ha cambiado entonces el período es un múltiplo de la duración especificada por la etiqueta EXT-X-TARGET-DURATION. Esta etiqueta indica el máximo valor EXTINF de cada fichero de media que puede agregarse a la lista de reproducción y es constante a lo largo del fichero de manifiesto. Otras características importantes son la actualización de del manifiesto para la generación en directo de los segmentos y los segmentos suelen durar 10 segundos. [17] [58]

### 2.1.5 Caso Streaming Adaptivo tipo Pull

En el caso que los clientes hagan uso de streaming adaptivo tipo Pull, se tiene como requisito mínimo que el servidor responda a las solicitudes HTTP GET. El cliente obtiene el fichero de manifiesto que identifica los ficheros que contiene la representación del stream en las diversas tasas de transmisión (calidades). El cliente obtiene el contenido a través de los fragmentos haciendo uso de una o varias conexiones de acuerdo al estado del buffer y las condiciones de red. [10]

El cliente necesita un fichero llamado manifiesto del cliente o una lista de reproducción para mapear las solicitudes de fragmentos a los ficheros respectivos. Los ficheros de los fragmentos estarán identificados por su ubicación temporal dentro del stream. En algunas implementaciones del streaming adaptativo se suele tener un manifiesto en el servidor.

La administración del buffer es una función muy importante que debe tener todo cliente de streaming adaptativo para poder seleccionar los fragmentos de un fichero a una tasa de específica dependiendo de las condiciones de red. Típicamente un cliente de streaming adaptativo mantiene un buffer de algunos segundos (de 5 a 30).

Un medio de transporte se requiere para comunicar las solicitudes del cliente al servidor. En el esquema de streaming adaptativo de tipo pull se hace uso de HTTP GET hacia un servidor WEB HTTP. También se pueden hacer uso de APIs de servicios web

(como las aplicaciones de Smooth Streaming Transport que se ejecutan en el servidor de Microsoft Internet Information Services). Las solicitudes GET usan una sola conexión TCP pero algunas implementaciones de streaming adaptativo hacen uso de varias conexiones TCP para solicitar varios fragmentos al mismo tiempo o para pedir audio y vídeo en simultáneo.

El cliente esta preconfigurado para solicitar contenido con cierta tasa de bits o perfil basado en los resultados de la monitorización de la red. Cuando un perfil es seleccionado y el cliente encuentra el URI establece una o varias conexiones al servidor. Hemos observado que distintos reproductores usan estrategias distintas. Algunos solo usan una conexión TCP para las solicitudes GET mientras que otros van abriendo y cerrando múltiples conexiones TCP durante una sesión de streaming adaptativo. Mientras el cliente mide el estado de su buffer decidirá si pide fragmentos de un perfil con mayor o menor tasa de bits. Algunas aplicaciones de streaming adaptativo abren una nueva conexión cuanto cambia a un perfil con mayor/menor tasa de transmisión.

### 2.1.6 Codificación de Video Escalable (SVC)

Un método alternativo de adaptación de la tasa de transmisión desarrollado por el método streaming, basado en alternar entre diferentes codificaciones del contenido o entre fragmentos de distintas calidades [19], es el SVC (Scalable Video Coding o codificación de vídeo escalable) [48]. El SVC permite a los clientes seleccionar los flujos de vídeo adecuados a las condiciones de la red y a sus posibilidades de decodificación. El SVC ha sido estandarizado como una extensión al estándar H.264/MPEG4 AVC. El SVC solo se aplica a flujos de vídeo (el audio no está considerado).

En SVC el flujo de bits del video está compuesto por una estructura de capas. La capa base tiene el menor nivel de calidad (resolución, cantidad de imágenes por segundo). Cada capa superior a la capa base provee mejoras de calidad. En consecuencia podemos mejorar o deteriorar la calidad solicitando más o menos capas de acuerdo a las condiciones de red o características de los dispositivos [48]. Las mejoras de calidad "knobs" en SVC se refieren a escalabilidad temporal, espacial y de SNR (la relación señal ruido, pues siempre queremos que haya más señal que ruido). La escalabilidad temporal da la posibilidad de agregar o quitar imágenes completas del flujo de vídeo. En su forma más básica los servidores que hacen uso de la metodología push la utilizan mediante el "stream thinning". El stream thinning consiste reducir la tasa de bits del flujo de vídeos, una manera de lograr esto es por medio de la eliminación de tramas de imágenes. La escalabilidad espacial codifica la señal de vídeo en varias resoluciones. El cliente puede utilizar las imágenes en baja resolución para predecir imágenes de mayor resolución con la información adicional en las capas superiores. La escalabilidad SNR codifica la señal

a una sola resolución pero en diferentes niveles dependiendo de la precisión solicitada. Cada capa de mejora incrementa la precisión de las capas inferiores.

Una ventaja de SVC está en la habilidad de distribuir la información en varias capas con poca información redundante entre ellas. En otras palabras mientras un stream (flujo) que esta codificado de manera tradicional en diferentes calidades tiene bastante redundancia entre codificaciones, en SVC se tienen muy poca información en común entre las diversas capas. Esto hace de SVC una manera muy eficiente de almacenar contenido multimedia. Otra ventaja de SVC es la degradación transparente sin la necesidad de la participación del cliente o el servidor cuando los paquetes se pierden (pues si se pierden las capas superiores solo hay una degradación del contenido no una pausa). Esto entra en contraste con la codificación por tasas de transmisión que requiere realizar un cambio en las calidades (una decisión que se tomará en el cliente o en el servidor). Los flujos SVC son más complejos en generar y conllevan limitaciones en los códecs de vídeo que se pueden utilizar, esto ha llevado a que SVC se vaya incorporando al mercado de manera más lenta.

### 2.1.7 Protocolos para Streaming Push versus Pull

A continuación realizamos una comparación de los protocolos para streaming Push y Pull. Una de las principales diferencias entre los esquemas basados en push y pull es la complejidad del servidor. En el caso de streaming pull la tarea de la administración de los flujos de datos es tarea del cliente lo que simplifica la tarea del servidor. Adicionalmente las implementaciones tipo pull pueden (y de hecho usan) HTTP. Como consecuencia un servidor web normal y con configuración estándar puede ser utilizado para brindar contenido multimedia a los usuarios. En contraposición para poder brindar contenido a través de un streaming tipo push se requiere un servidor que tenga implementado el protocolo RTSP o similar para poder realizar las tareas de administración de bitrate, retransmisión y cacheo del contenido. Esto hace que las implementaciones pull sean más baratas que las push. Sin embargo el streaming tipo pull es menos eficiente, y tiene demasiada sobrecabecera. La sobrecabecera de HTTP con TCP es mayor que RTP con UDP teniendo en cuenta que la mecánica de retransmisión y control de congestión no existen en RTP. Ambos estilos de streaming permiten al cliente tener un buffer y tener funcionalidades como el 'trick-mode', como 'play' y 'fastforward'. Estas funcionalidades se realizan en el cliente para evitar el buffer 'underflow' (cuando ya no se tiene que reproducir debido a que la tasa de transmisión es menor que la velocidad de reproducción) y tener una reproducción de contenido fluida. En el streaming adaptativo durante el inicio el cliente puede ser no adaptativa (al bajar contenido de baja calidad) para mejorar la velocidad de arranque y reducir el tiempo de respuesta del sistema (con contenido en baja calidad el vídeo comenzará a

reproducirse de manera más rápida). Uno de los mayores beneficios del streaming en push es el soporte para multicast. Multicast permite a los servidores enviar un solo paquete a un grupo de clientes que esperan recibir dicho paquete. El paquete se distribuye de manera eficiente en la red y los usuarios puede unirse o salir de un grupo de multicast. En el streaming tipo pull se establecen conexiones unicast es decir uno a uno entre el servidor y el cliente. En el esquema unicast se tienen tantos caminos como clientes se tengan por lo que hay que enviar el mismo paquete a todos los clientes de manera individual. Asimismo, los nodos intermedios reenvían los mismos paquetes. Todo esto hace que el sistema unicast reduzca la eficiencia de la red. La distribución y el encaminamiento de los paquetes a través de la red podrían hacerse más eficiente haciendo uso del cacheo de contenidos. El contenido es cacheado a lo largo de la red en servidores de cacheo distribuidos. Con este sistema un usuario puede obtener el contenido que busca de un servidor de cache en vez de la fuente directa. La manera más eficiente de usar el caché es mediante un streaming tipo pull donde cada fragmento puede ser almacenado en la red de manera independiente.

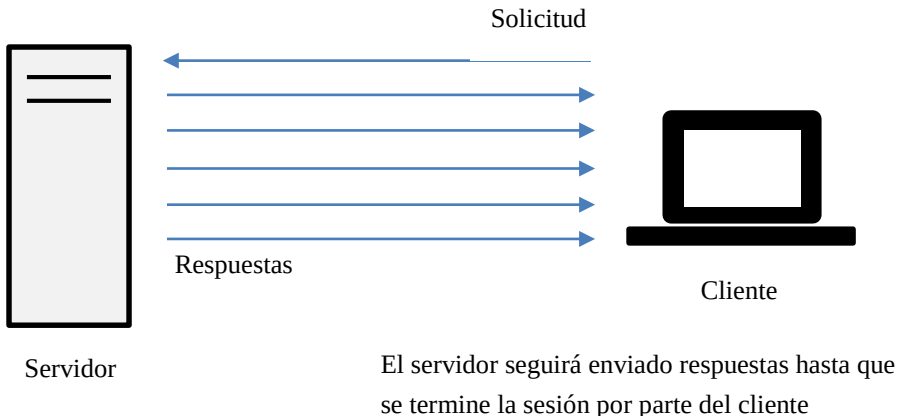


Figura 10. Protocolo tipo push

### 2.1.8 Aplicaciones para Realizar Streaming

Existen muchas aplicaciones de streaming de vídeo en la Internet. Las aplicaciones más importantes están en los sectores de dispositivos móviles y entretenimiento en casa. Las calidades de vídeo para esos sectores son PD (portable), SD (standard) y HD (high definition). Entre las aplicaciones se encuentran las que pertenecen al entorno cerrado de los ISPs y las aplicaciones de Internet. También pueden operar en diversas redes como las administradas. En los siguientes apartados se comparan diversas redes y el uso de streaming tipo pull y push.

**Streaming en Dispositivos Móviles**

Los usuarios consumen streams de películas en Youtube y otros contenidos cuando están movilizándose, en la universidad, hoteles, etc. La codificación usada para dichos casos es la PD que es una codificación con menos resolución y tasa de bits que SD y HD. Se distribuye video en calidad PD es debido a las velocidades más lentas y con mayores pérdidas de paquetes que son comunes en las redes móviles. Variaciones en la carga de las celdas de la red, en la radio propagación y fluctuaciones en la capacidad de la red hacen que las aplicaciones de vídeo tengan que adaptarse a las distintas redes. Entre las redes que se tiene que adaptar se incluyen: GPRS, EDGE, UMTS, CDMA y LTE por lo que se entiende el interés de los operadores de red por los estándares de streaming adaptativo. Cuando la conexión se realiza con un servidor en una red cerrada (walled garden), el proveedor puede administrar la distribución del vídeo a un cliente. Si el cliente se encuentra en la Internet entonces no se puede garantizar la calidad de servicio. Un tercer caso se da cuando se combina distribución móvil en hogares.

**Walled Garden - Red Privada del ISP**

En caso de haber una red propietaria toda la conexión punto a punto se encuentra en la red del proveedor de servicio. El Operador de red puede optimizar su red para la distribución de vídeo. El proveedor puede usar escoger usar streaming adaptativo tipo push para usar de manera eficiente su red y mejorar la calidad de vídeo en diversas condiciones de red. Alternativamente un proveedor de Internet puede reutilizar sus servicios de red como servidores Web HTTP y caches para desplegar un servicio basado en streaming tipo pull.

**Internet Móvil**

Con internet móvil el proveedor de servicio ya no puede administrar la calidad punto a punto cuando la conexión atraviesa el Internet público. En este caso los servicios basados en streaming pull tienen ventaja como indicaremos a continuación.

**Caso de Uso Hogar-Móvil**

En este caso el ISP de Internet móvil no puede garantizar un servicio cuando los primeros y últimos saltos se encuentran en la red del hogar del usuario. Redes que hacen uso de femtoceldas y servicios de redes caseras (home-network) a usuarios móviles son ejemplos. Cuando una red no-administrada es parte del camino en la conexión punto a punto y en particular cuando es un salto inalámbrico los streamings tipo pull son preferibles a los métodos push.

**Streaming en Redes Domésticas**

Los casos de uso dentro del hogar son importantes debido a que los usuarios suelen consumir streams de vídeo en casa. En muchos hogares la red consiste simplemente en un solo cable Ethernet conectado a un modem de banda ancha, redes más complejas están comenzando a hacerse populares en varios países.

Las redes WIFI son utilizadas en muchas regiones. Un problema muy estudiado en la distribución de vídeo son las variaciones de velocidad y confianza en red debido a su topología. Artefactos electrodomésticos, objetos de metal y otras fuentes de interferencia hacen que un televisor en una habitación no pueda proveer la calidad de otro televisor en otra habitación. Las diferentes capacidades de red en los equipos conectados a las redes 802.11 causan dificultades con el uso del protocolo multicast. El vídeo bajo demanda en lugar de las transmisiones de televisión será la forma predominante de vídeo en las redes caseras. Desafortunadamente las redes inalámbricas y otras redes comúnmente usadas en el entorno doméstico suelen ser vulnerables a la interferencia y sobrecarga. La utilización de la frecuencia de 5GHz en vez de la de 2.4GHz mitiga parte de la interferencia en las redes WIFI sin embargo los resultados son variados respecto a la velocidad de red en la red inalámbrica. Las comunicaciones por PLC (power line communication) también son vulnerables a la interferencia. De hecho una posible excepción para un buen servicio en casa sería tener una red Ethernet cableada con cables de categoría 5. El streaming adaptativo es muy importante para el entorno doméstico donde las redes no son mantenidas, controladas, administradas y tiene una gran variedad de posibilidades de velocidad. Sería raro que un usuario seleccionará la calidad de vídeo acorde con su velocidad de red. El vídeo en redes domésticas con administradores de red no será común en los hogares aunque seguirá brindando una calidad de vídeo mejor a los hogares que las redes no-administradas solo pueden intentar replicar. A continuación se consideran redes administradas y 'silvestres' y comparamos la aplicación del streaming push y pull.

**Streaming a un Cliente en el Hogar desde un Servidor Doméstico**

Este caso es el más interesante sin embargo es el menos común. Por años usuarios entusiastas han conectado su TV a sus redes para reproducir películas desde sus ordenadores, jugar con sus consolas o conectarse con un servidor, pero esto todavía está lejos de ser lo usual. En Estados Unidos las redes Wi-Fi pueden conectarse a un servidor, a un PlayStation y otros dispositivos que implementan la especificación DLNA (Digital Living Network Alliance). Sin embargo los usuarios sin conocimientos técnicos no usan estas capacidades. Configurar un servidor de multimedia doméstico requiere que el usuario tenga conocimientos de como cargar películas comerciales con reproductores que puedan decodificar contenidos con DRM (Digital Rights Management) y tener los

códecs para los distintos formatos. Dichas tareas están fuera de la rutina de los usuarios domésticos. Sin embargo los servidores en casa pueden ser bastante útiles para aquellos que desean evitar los cuellos de botella de la Internet (y de las redes de los ISP). Con los servidores domésticos los usuarios pueden tener acceso al contenido cuando la red del ISP esta fuera de servicio y reproducir el contenido en todos los televisores y dispositivos del hogar. iTunes ha implementado el uso compartido de música dentro de los hogares. Apple y otros vendedores están desarrollando el uso compartido de vídeo entre dispositivos conectados a la red doméstica. Soluciones estándar como DLNA abren el mercado a una mayor cantidad de productos y son importantes para el futuro de los casos de uso del vídeo en casa. El servidor de contenidos en casa es sin duda más importante para el futuro que para el presente. Sin importar si se usan estándares o reproductores privados (propietarios) las redes inalámbricas en el hogar están limitadas por el ancho de banda, el ruido y la interferencia por lo que la red doméstica es el eslabón más débil de la cadena de distribución de contenidos. La calidad de vídeo en una red inalámbrica doméstica sufre de todos los problemas de una red no-administrada más la interferencia, ruido, pérdida de paquetes y uso compartido de recursos con otros elementos de la red. Cuando los dueños de la redes domésticas hacen uso del streaming adaptativo tipo push deben instalar y configurar un servidor streaming especializado en una red que no está administrada por una empresa como un ISP. Los protocolos más populares para distribuir vídeo son HTTP seguido por RTP (Real-time Transport Protocol) y RTSP (Real-time Straming Protocol). HTTP se usa en el Web y en protocolos domésticos como DLNA. [11]

**Streaming desde un Servidor de Internet al Hogar**

En este caso se incluyen YouTube, Hulu, iTunes y otros contenidos de Internet que se distribuyen a los ordenadores o Smart TV. Como se explicó anteriormente el último salto en el hogar donde hay una red inalámbrica pude ser problemático. El Internet solo soporta la distribución de vídeo sin mecanismos de control (unmanaged). Por lo que el streaming tipo pull es el más adecuado para este esquema. [11]

**Streaming a un Usuario en el Hogar desde una Red de Pares P2P (Peer-to-Peer)**

En ciertas partes del mundo como en Asia y sectores de Europa es muy popular usar el streaming P2P. La investigación ha demostrado que P2P puede reducir la carga a los servidores y proveer mejor escalabilidad para numerosos usuarios que hacen uso del streaming. Sin embargo mejoras en los diseños de los servidores y las estructuras de cacheo permiten buenas capacidades de streaming sin necesidad de hacer uso del P2P. Desde el punto de vista del negocio la falta de control sobre la distribución P2P falla en brindar ganancias entre los proveedores de contenido y los ISPs. A pesar de que se están

haciendo esfuerzos de cara a hacer el transporte de contenidos confiable en P2P todavía no hay modelos de negocio. [11]

### 2.1.9 Casos de Uso de Streaming Adaptativo Tipo Push

De los casos presentados muchos tienen que atravesar redes sin administración en su recorrido desde el servidor hasta el usuario final. Hay dos casos que se benefician del streaming tipo push que son los casos de servidores administrados por el ISP dentro de su red (sin salir al Internet comercial). Los ISP están optimizando sus redes para la distribución de vídeo en HD. Las redes propietarias de los ISP cumplen con los requisitos de seguridad que los productores de vídeo solicitan. Adicionalmente los proveedores del servicio de vídeo tienen relaciones estratégicas de negocios que les permiten tener un espacio en los estrenos del contenido a su vez que la posibilidad de realizar eventos pay-per-view. Esto da a las empresas que brindan el servicio un lugar importante en la distribución de películas populares. Las redes en Internet tienen las redes propietarias como referencia y meta. Es importante analizar que tanto las redes de Internet se pueden aproximar a las redes propietarias. El Internet da acceso a la mayor cantidad de servicios (YouTube, iTunes, Netflix, Hulu, etc). Si el streaming tipo pull puede llegar a ser tan bueno como las redes propietarias entonces será el sistema de distribución por defecto. De los casos restantes realizar streaming desde un servidor en casa hacia otros dispositivos es mucho menos común que usar un servicio de streaming como YouTube o Netflix. Este caso se volverá más importante cuando los ficheros en discos duros sean reemplazados por ficheros en cloud. Los servidores domésticos cobran importancia cuando las redes de banda ancha no están disponibles debido a que los usuarios podrán tener acceso por lo menos a los contenidos en su servidor doméstico. Dicho caso solo se enfoca en la red doméstica que puede ser 802.11 y es de gran interés para los servicios en Internet como a los de redes propietarias [11].

### 2.1.10 Esfuerzos de Estandarización

El formato PIFF (Protected Interoperable File Format) [46] y el SST (Smooth Streaming Transport) de Microsoft son especificaciones completas para streaming adaptativo. PIFF y SST hacen uso de mp4 y de ficheros fragmentados. Microsoft también ha hecho estos protocolos abiertos para su uso masivo al ofrecer una licencia a través de Microsoft Community Promise. Apple también ha ofrecido su tecnología de HTTP Live Streaming libremente en la Web y como un borrador de la IETF. El 3GGP fue la primera organización en publicar una especificación para streaming adaptativo sobre HTTP. Fue publicado en el Release 9 en 2010. Con la iniciativa de MPEG se aprobó el primer draft de DASH (Dynamic Adaptive Streaming over HTTP) en el 2010.

Dentro de este campo todavía hay que estudiar los efectos del control de congestión, ACK retardados en la distribución del vídeo y en los protocolos utilizados. Hay que

analizar si es mejor tener muchas conexiones TCP a utilizar el protocolo SCTP (Stream Control Transmission Protocol)

## 2.2 Protocolo de Internet para Television (IPTV)

IPTV se puede definir como servicios multimedia como video, televisión, audio, texto, gráficos y datos distribuidos a través de una red IP. IPTV debe tener una administración (por un operador de IPTV) para poder ser confiable, seguro, interactivo y alcanzar buenos niveles de calidad de servicio. En este sentido IPTV se asimila a los servicios de televisión por cable o satélite que tienen un equipo de usuario a la cual el usuario conecta sus televisores. La característica principal es que la arquitectura de red del operador será IP y los usuarios deberán tener suscripciones con los proveedores de servicio para tener acceso a los distintos canales. Con respecto a la estandarización la ITU se encarga de mantener la documentación para tener una IPTV sobre una misma plataforma y que de esta manera se interoperable (equipos hardware, software proveedores de contenidos).

### 2.2.1 Estándar H.720 sobre terminales finales IPTV

El estándar H.720 definen los dominios de los terminales IPTV y los sistemas finales. Para poder implementar un servicio de IPTV se definen dominios que son: el dominio del usuario, el dominio del proveedor de red, el dominio del proveedor de servicio y el dominio del proveedor de servicio. Un ejemplo de estos dominios se muestra en la figura 11. [6]

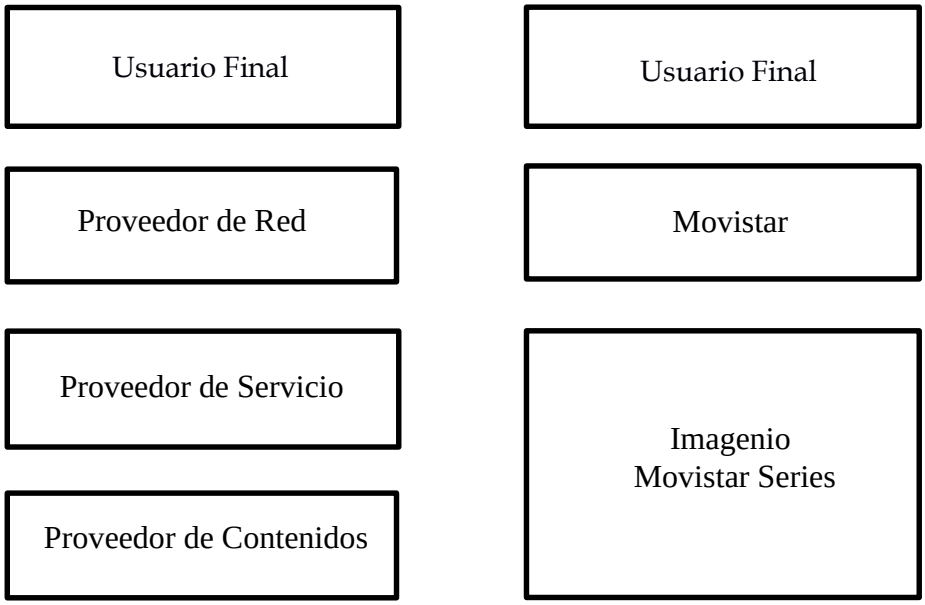


Figura 11. Servicio IPTV y dominios

La arquitectura de un servicio de IPTV consistirá de funciones de usuario final, las funciones de las aplicaciones, el control de servicio, la distribución del contenido, las funciones de red y las funciones administrativas.

Las funciones de usuario tienen dos aspectos muy importantes entre los que se encuentran: las funciones del terminal y las funciones de red del hogar. Las funciones del terminal se definen como aplicaciones de cliente, funciones de protección de servicios y contenidos (SCP), funciones de distribución de contenido y funciones de control. Las funciones de red hogar son la de pasarela para las funciones de distribución de contenido.

Las funciones de aplicación están conformadas por las aplicaciones de IPTV, las funciones de perfiles, las funciones de preparación de contenidos y las funciones SCP. Las funciones de distribución de contenido tienen las funciones de localización, control de la distribución del contenido y almacenamiento. Las funciones de control de servicio tendrán las funciones del servicio de IPTV y del perfil de servicio del usuario.

Las funciones de red estarán implementadas mediante el uso de las funciones de autenticación y control de recursos también se tendrán un sub bloque con funciones de transporte como el acceso a la red, las funciones del núcleo y la frontera de la red.

Las funciones administrativas tienen la administración de la aplicación, el control del servicio, la distribución del contenido, el manejo del terminal del usuario, el control del transporte. Adicionalmente hay funciones de proveedor de contenidos. Todas estas funciones se muestran en la figura 12 obtenida de a especificación de la ITU-T.

La arquitectura de IPTV está planteada para brindar una gran cantidad de servicios entre los que se encuentran los servicios de distribución de contenidos que pueden ser: servicios bajo demanda, tradicionales (como la televisión tradicional), de publicidad, contenidos suplementarios y almacenamiento de contenido para ser visto luego o desde otra ubicación. Los servicios interactivos podrían consistir de servicios de información, comercio, entretenimiento, aprendizaje, consultas médicas, monitorización, navegación web y publicidad interactiva. También se incluyen servicios de comunicación y otros de interés público como sistemas de alerta de emergencias.

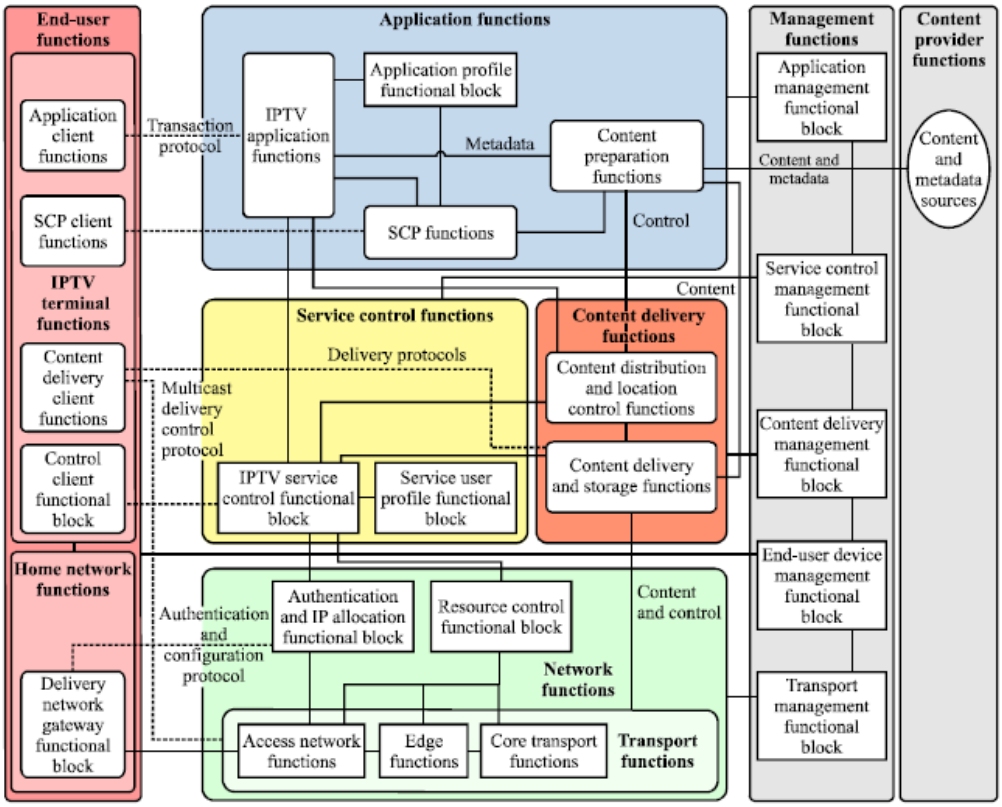


Figura 12. Arquitectura IPTV

Dentro de la implementación de acceso a contenidos grabados o almacenados por parte de los usuarios se plantea almacenar el vídeo en los mismos dispositivos, en la red del hogar o en una combinación de ambos. Esta forma de implementación es una forma de tener un servidor de contenidos en casa, lo que puede ayudar a la eficiencia de la red al descargar los ficheros multimedia en horas de poca carga para la red.

### 2.2.2 Estándar H-760 sobre lenguajes de aplicaciones en IPTV

El estándar H-760, estandariza la plataforma para declarar aplicaciones multimedia para IPTV. La plataforma de declaración de aplicaciones es una plataforma sobre el cual aplicaciones escritas usando un lenguaje de marcado (con la posibilidad de correr scripts tipo ECMAScript) pueden ejecutarse. Los lenguajes que permiten declarar aplicaciones en IPTV son Formato Binario para Escenas (BIFS, Binary format for scene) que es parte del estándar MPEG-4, el lenguaje de marcado de Broadcasting (BML, Broadcasting markup language), el CEA-2014 que permite ejecutar visualizar una interfaz de usuario de manera remota, también se pueden utilizar, CSS, DOMs, DVB-HTML que permite a los televisores acceder a la Internet. Otros lenguajes que también se pueden utilizar en IPTV son: ECMAScript (JavaScript), HTML, XHTML, SVG entre otros. Lo que da una gran versatilidad a las aplicaciones que se pueden implementar sobre IPTV. [7]

### 2.2.3 Estándar H-770 sobre mecanismos de descubrimiento IPTV

El estándar H-770 estandariza los mecanismos para el descubrimiento de servicios IPTV. Los mecanismos de descubrimiento de servicios IPTV empiezan a partir de un punto de entrada que dependerá de la arquitectura de IPTV usada. Un punto de entrada puede ser suministrado a partir de un servicio de información de proveedor de IPTV. El proveedor del servicio podrá ejecutar una solución basada en navegación web o metadatos o ambas. Adicionalmente la información de los servicios se podrá realizar mediante un algoritmo tipo pull o push. En el modo push la función de selección y descubrimiento de aplicaciones y servicios (service and application discovery and selection, SADS) activamente envía información de descubrimiento de servicios y aplicaciones. En el modo pull el terminal IPTV solicita periódicamente solicitudes de servicios y aplicaciones al bloque funcional SADS.

La información acerca de los proveedores de servicio contendrá el identificador del proveedor de contenidos, una descripción del servicio, el nombre del proveedor y un URL donde encontrar el servicio entre otros. Los mecanismos de transmisión de esta información será a través de HTTP, HTTP sobre TLS, IGMP v2, IGMP v3, FLUTE multicast en modo push, DVBSTP. Los servicios ofrecidos podrán ser, de televisión tradicional, en colecciones, guías de contenidos y servicios de otros proveedores.[8]

## 2.3 Redes de Distribución de Contenidos (CDNs)

Las redes de distribución de contenido (CDNs) son muy importantes en el funcionamiento de la Internet. Hay varios tipos de diseño de las CDNs uno de los cuales se le podría nombrar “enter deep” y la otra forma como “ISP to home”. Las redes de distribución de contenidos (CDNs) dan acceso a la diversa información multimedia en distintos lugares a lo largo de muchas redes de proveedores de Internet y en los puntos de presencia (POPs). Los servidores de una red de distribución de contenido operan de manera transparente para brindar una mejor experiencia de usuario. Cuando un cliente hace una solicitud la CDN escoge un servidor cercano para optimizar la experiencia de usuario percibida. Los servicios más tradicionales de las CDNs son el despliegue de páginas web estáticas y descargas de ficheros grandes (como actualizaciones y parches). Las CDNs también proveen aceleración de las aplicaciones al brindar contenidos dinámicos, soportar comercio electrónico, bases de datos, seguridad SSL y aplicaciones de Web 2.0. Las CDNs también ayudan en la experiencia de los usuarios al brindar servicios de contexto y ubicación. Las redes de distribución de contenido proponen que son la solución para la distribución de vídeo streaming. La infraestructura de las CDNs que consiste en miles de servidores a lo largo de la red es una parte importante de la red.

El diseño de CDN tipo "enter deep into ISP" es desplegar los servidores de distribución de contenidos en los puntos de presencia (POP) de los ISP. La idea es acercarse a los usuarios finales y proveer mejoras en el desempeño del retardo y el ancho de banda. Este tipo de diseño resulta en un gran número de servidores esparcidos por el mundo. Como consecuencia de este diseño altamente distribuido la tarea de operar y mantener la red es compleja. Dentro de la complejidad se incluye los algoritmos para repartir los datos a los servidores a través del Internet comercial. Un ejemplo de dicho diseño es la red de Akamai. [12]

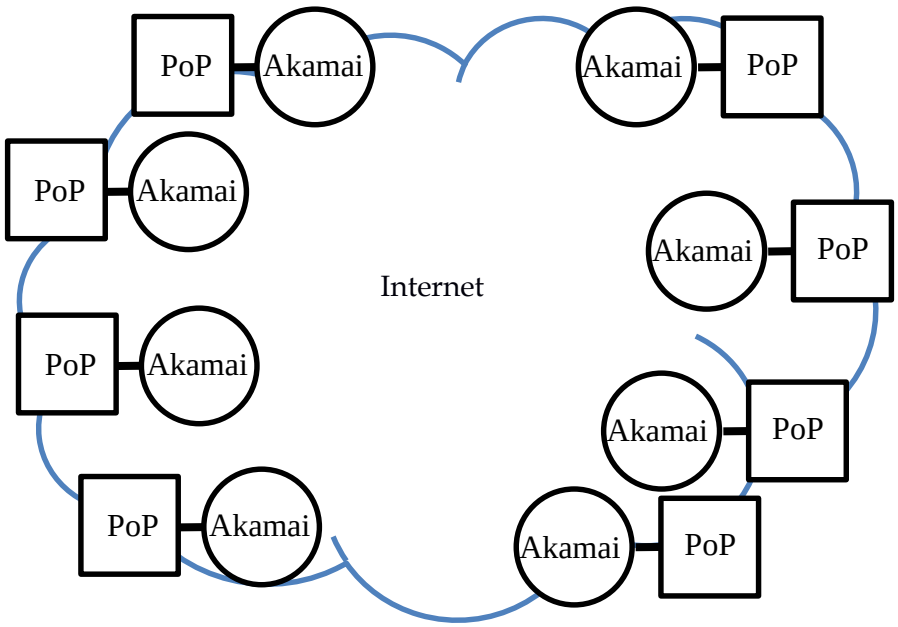


**Figura 13. Arquitectura altamente distribuida "enter deep into ISP"**

La otra filosofía de diseño es traer el ISP a los centros de datos que consiste en disponer de centros de datos grandes en ubicaciones cercanas a los puntos de presencia (PoPs) y conectar estos centros a través de conexiones de alta velocidad a los PoPs. Para este tipo de diseño la CDN se ubica lo más cerca posible de los POPs de los ISPs. Este diseño tiene una operación y mantenimiento menor que el anterior pero a costa de posiblemente tener un poco más de retardo a los usuarios finales. Limelight usa este tipo de arquitectura.

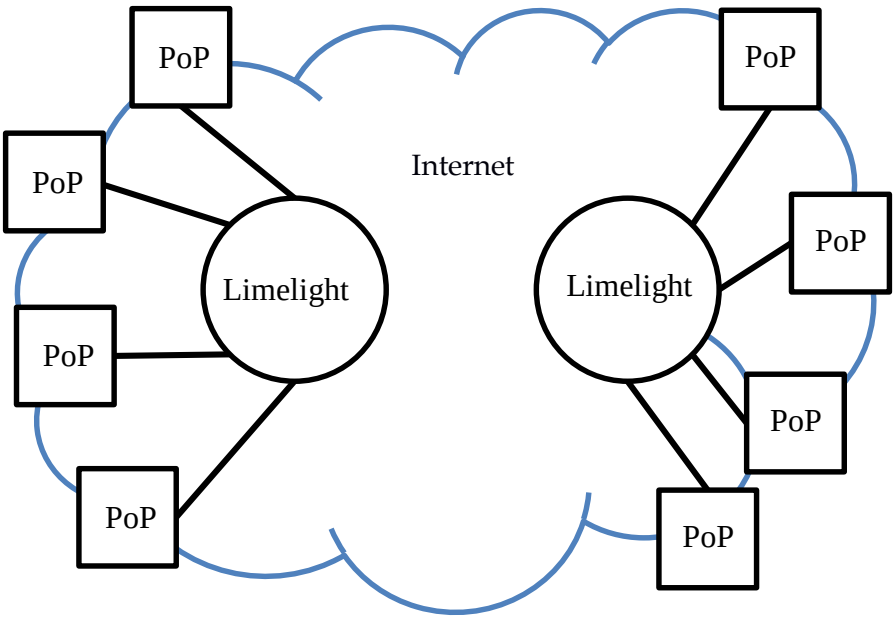


Figura 14. Arquitectura CDN concentrada "bring ISPs to home"

Uno de los problemas que motiva el uso, estudio y optimización de las redes de contenidos son las proyecciones del incremento del tráfico en las redes por el uso del streaming de vídeo y la falta de recursos para aumentar los recursos de red para afrontar dicho crecimiento.

**Estructuras de las CDNs**

La infraestructura de red de una CDN consiste mayormente de una red de contenidos y una red de servidores DNS. En la red de Akamai había aproximadamente 27000 servidores mientras que en Limelight había 41000. Adicionalmente Akamai tenía cerca de 6000 servidores DNS mientras que Limelight tenía alrededor de 3900. Las fuentes principales del retardo son dos: el tiempo de resolución del DNS (que es el tiempo que se toma en encontrar la mejor fuente) y el tiempo de dicha fuente al usuario. Las CDNs de gran escala usan un gran número de servidores DNS que están en la misma ubicación que los servidores de contenidos. Como referencia se tiene que el retardo en la búsqueda del servidor DNS de Akamai era de 138ms contra los 170ms de Limelight también los servidores de contenido de Akamai (103ms) eran más rápidos que los de Limelight (222ms).

Cuando un usuario solicita una página primero pide la IP a su DNS local si este no lo tiene este a su vez hace su solicitud a un DNS Primario que le devuelve un CNAME si el servidor de DNS local no tiene la IP del CNAME este volverá a consultar al DNS primario por la IP del CNAME de la página buscada. [12]

| | Tipo | # de CNAMEs |
|---|---|---|
| a | *.akamai.net | 1964 |
| b | *.akadns.net | 757 |
| c | *.akamaiedge.net | 539 |

**Tabla 8: Número de CNAMEs**

Akamai tenía cerca de 3260 CNAMEs. Los CNAMEs de Akamai pueden clasificarse en tres tipos y con distintas funcionalidades. El primer tipo de servicio, akamai.net, está de acuerdo a la noción de una red de distribución de contenido, cada resolución de DNS devuelve 2 o más direcciones IP. Cuando se realizan las consultas de distintas ubicaciones se obtienen diferentes IP. Para cada CNAME había cientos de direcciones IP y se encontraron aproximadamente 11500 direcciones IP únicas. El segundo tipo de servicio es el de akadns.net que se usaba para para realizar balanceo de carga en usuarios que tienen sus propios servidores de contenidos. El último tipo de servicio, el de akamaiedge.net, se utilizaba para realizar una distribución de contenido dinámico. Cabe mencionar que Akamai usa tecnología de virtualización para dar a sus usuarios entornos aislados entre sí. La estructura de Akamai se basa en dos niveles de DNS. Entre otros datos se tiene que Akamai tendría como 27000 servidores de contenidos de los cuales 6000 tienen servidores DNS. Más del 60% de los servidores están en los Estados Unidos sin embargo casi el 90% están en ISPs que no son de Estados Unidos.

La red de Limelight parece solo tener un nivel de DNS y usa IP anycast para anunciar el nombre de los servidores. De los aproximadamente 4100 servidores de contenido de Limelight cerca de 3100 también tienen servidores DNS. Las ubicaciones de los clusters de Limelight son de 10 en Estados Unidos y 9 en otras partes del mundo.

Adicionalmente las CDNs deben poder tener resistencia a los fallos porque estos dañan al negocio y a la confianza de los clientes. Debido a la naturaleza menos dispersa de la arquitectura de CDN de Limelight (esto bring ISP to home) podría verse más afectado por los fallos de uno de sus nodos. Al realizar pruebas de fallos en los centros de datos de Limelight se concluye que Limelight es bastante robusto ante la falla de los centros de datos, esto es debido a que el incremento del retardo es poco. El retardo llega a aumentar de 142ms a 150ms (un 5.5%) cuando fallan 4 centros de datos. El máximo retardo encontrado para el fallo de 4 servidores fue cuando el retardo aumentó de 142ms a 182ms (28.2%) pero esto fue debido a que los 3 servidores de Asia no estaban disponibles ni el de San José que sería un caso extremadamente raro. [12]

### 2.3.1 Principios de Diseño

A continuación se presenta los principios de diseños de redes de distribución de contenidos altamente distribuidas. Para una red de distribución de contenidos altamente distribuida se proponen usar como criterios de diseño: la ubicación óptima de los servidores de caché, los mecanismos para el enrutamiento de las solicitudes, distribución de contenido duplicado y subcontratación de servicios. Un aspecto que no se tratará mucho será la adquisición del contenido tiene que ver con el mecanismo mediante el cual los datos ingresan en la CDN (por ejemplo mediante el uso de múltiples formatos con distintos códecs). Las demás consideraciones de diseño tienen que ver con el emplazamiento de los servidores caché, distribución del contenido, lógica del procesamiento (y enrutamiento) de las solicitudes y la administración de la CDN. [13]

**Ubicación de los Servidores de Caché**

Una de las principales decisiones de diseño de un operador de CDN son el emplazamiento físico de los servidores de caché. Los servidores de caché pueden ubicarse desde la red de acceso de radio al núcleo de la red. Su ubicación afecta tanto el CAPEX (capital expenditures) como el OPEX (operational expenditures).

Se puede mencionar que existen varias estrategias para ubicar los servidores de caché entre los que se encuentran el problema del mínimo número de centros (minimun k-center problema) y la separación jerárquica k-HST (k-hierarchically well separated trees), sin embargo dichos métodos son difíciles de implementar por lo que se hace uso de métodos heurísticos para encontrar las ubicaciones de los servidores de red.

En el caso del uso de rede móviles se deben ubicar los servidores de caché donde se pueda acceder con un mínimo de tiempo de acceso y retorno (RTT). Es debido a esta consideración que es importante considerar la infraestructura de red móvil existente. Por estos motivos se considera implementar los servidores en los niveles de enrutadores fronteras como en el nivel de acceso a la red de radio. [11]

**Externalización del Contenido**

La distribución del contenido es una de las principales funciones de una CDN que consiste en ubicar y externalizar el contenido (la estrategia de externalización de replicar los contenidos en servidores de caché). La estrategia para la externalización del contenido puede ser de tipo push o tipo pull bajo demanda. En el caso tipo push debe haber un algoritmo para decidir cuándo se realiza la inyección del contenido, en el caso tipo pull el contenido se almacena solo ante la solicitud de un usuario. El impacto de las dos estrategias es el de la latencia percibida por el usuario y de la complejidad de los mecanismos que intervienen.

Las CDN más populares (como Akamai) usan un sistema no-cooperativo tipo pull. El compromiso en utilizar esta estrategia es que el mejor servidor no siempre es seleccionado. Por otra parte el enfoque cooperativo tipo pull permite a los servidores de caché cooperar entre sí (al estilo P2P) un ejemplo de esto es la CDN académica Coral que hace uso de una variación de las tablas distribuidas de hash (DHT)

Una recomendación que se hace es la de usar ambas estrategias push, pull y adaptativas para usar el esquema apropiado en base a la popularidad del contenido y sugerencias de los proveedores de contenidos. Un esquema de fragmentación con mucha granularidad es un elemento esencial para permitir la flexibilidad en la externalización de los contenidos.

**Administración de Contenidos no Replicados**

Otra de las funciones prioritarias es la obtención del contenido (mover las réplicas entre los distintos servidores de caché) esta función tiene que responder al problema de cómo obtener el contenido cuando el servidor de caché local no tienen réplicas. Una propuesta es usar un esquema cooperativo como el P2P que consigue el contenido de varios servidores de caché, la otra propuesta es obtener los datos de un servidor centralizado. Los factores a tener en cuenta a decidir son la posibilidad de soportar balanceo de carga y adaptarse a situaciones en las que hay sobrecarga. La mayoría de los algoritmos de replicación son soluciones que tratan de manejar las cargas, tamaños de los ficheros y retardo total de acceso. Entre las estrategias heurísticas de replicación mencionan: la aleatoria, popularidad entre otros. [49]

### 2.3.2 Selección de CDNs

La selección y distribución del contenido (la selección y la distribución del contenido adecuado del servidor de caché al usuario final) tiene una importancia fundamental dentro de una CDN. En este aspecto interviene el enrutamiento de las solicitudes del cliente a la CDN apropiada en los que intervienen criterios como la selección de métricas (como la proximidad en saltos de red o la latencia percibida) para tomar la decisión. Una propuesta es utilizar enrutamiento jerárquico [50].

Cisco ha implementado un algoritmo adaptativo de enrutamiento de solicitudes que tiene en cuenta una combinación de tres medidas (que tienen distintos pesos) que son: la distancia entre sistemas autónomos, de distancia al interior de los sistemas autónomos y el retardo extremo a extremo. Dicho algoritmo es costoso debido a la necesidad de implementar la monitorización en cada servidor y se genera un tráfico en la red. Akamai por su parte usa un algoritmo propietario que toma en consideración las cargas de los servidores, la confiabilidad en las cargas de los clientes con los servidores y el ancho de banda disponible para cada servidor.

Por último es importante reconocer que un factor relevante es la administración de la CDN en sí que permite a la CDN ser eficaz a la hora de brindar servicio a sus clientes. Para poder tener esta funcionalidad se debe poder obtener métricas para poder evaluar el desempeño de la CDN. Las métricas más comúnmente usadas son: el ratio de popularidad de cierto caché, el ancho de banda de los servidores, retardos, porcentaje de utilización de los servidores de caché y disponibilidad.

Es importante mencionar que el uso de costos en los enlaces de red ayuda a los algoritmos a maximizar el enrutamiento de las solicitudes. El balanceo de las solicitudes dependerá de los modelos y costos asignados a los recursos de red. Adicionalmente se propone un manejo descentralizado del enrutamiento. Esto propone responder a las solicitudes de manera local y solo realizar consultas a servidores de mayor jerarquía en caso estos no se encuentren presentes. Una forma es utilizar métricas calculadas de los mismos servidores para definir los costos y definir que servidor se podría utilizar como se propone en [51].

El caso que se presenta es el de una red de distribución de contenidos que se utiliza para brindar vídeo a sus usuarios. Dentro de toda la red hay un nodo central que procesa las solicitudes de los clientes. El cliente realiza una solicitud al nodo central y este lo redirige a uno de los nodos de datos disponibles. El vídeo estará en varios nodos por lo que el nodo central tendrá que escoger cuál es el más apropiado. Dentro de las consideraciones a tener se tienen que no se puede redirigir todas las consultas a un solo nodo debido a que el nodo se sobrecargaría y por la variabilidad de las condiciones de red entre el cliente y el nodo. Se sugiere el criterio de tiempo de descarga entre el cliente y el servidor como criterio de selección. Este criterio de tiempo de descarga no es un parámetro de la red en sí sino que se deriva de otros parámetros como el retardo y la pérdida de los paquetes. Por lo que se estudia como los parámetros de red afecta al tiempo de descarga. Otros criterios que no son de la red pero de los servidores mismos son: el tiempo de carga de los servidores (carga de CPU, memoria, capacidad de red). Una propuesta para estudiar los parámetros de red no es usar los parámetros de red directamente sino sus derivados como el ratio entre el tiempo para descargar el fichero y el tiempo para ver el vídeo. Cualquier ratio que sea menor a 1 se considerará bueno debido a que la descarga toma menos tiempo que ver el vídeo. El tiempo de visualización del vídeo es un valor que no depende de las condiciones de red sino que es intrínseco al vídeo en sí. El tiempo de descarga se puede descomponer en dos tiempos: el tiempo de procesamiento y el tiempo de transporte en la red. Se ha supuesto que los parámetros de red no oscilan mucho de la media y que los valores medios son periódicos (los valores se repiten en las mismas franjas horarias o diarias). Los nodos de la CDN se reportan al nodo central y así puede decidir que nodos son buenos o malos para atender a los clientes. Una propuesta es usar un modelo simplificado de transmisión TCP y

escoger la CDN que tenga menor pérdida de paquetes. Otra consideración muy importante es utilizar el nodo menos cargado para esto se evalúan el número de procesos en ejecución, uso de la memoria y uso de las interfaces de red. [14].

### 2.3.3 Redes de Distribución de Contenidos para Entornos Móviles

La distribución de contenido multimedia a usuarios de tipo móvil presenta retos debido a la transferencia de paquetes a usuarios con movilidad inalámbrica y gran capacidad de crecimiento. Para afrontar este reto se presenta una arquitectura de red de distribución de contenidos (CDN) diseñada para servir a usuarios usando dispositivos móviles. Los avances en tecnología móvil brindan banda ancha a los usuarios. Este acceso a la banda ancha genera aplicaciones que hacen uso de ella y crean la necesidad de tener una infraestructura que pueda afrontar los requisitos para realizar un streaming de tiempo real. El entorno móvil es muy dinámico, los canales son propensos a fallos y son muy escalables respecto al número de usuarios. Casos como la distribución de páginas web ha resultado difícil debido a la congestión de la red y sobrecarga de los servidores [15]. Adicionalmente hay otras consideraciones como calidad de servicio, el proceso de "handover" entre las estaciones de radio y la latencia asociada dicho proceso [52].

Las características de una red de distribución de contenidos streaming móvil son la interoperabilidad con la infraestructura actual, también debe ser flexible para permitir ajustes particulares para otros sistemas. En ese sentido, el sistema debe ser escalable, para poder cumplir dichos requisitos, y debe ser de naturaleza modular. Otro requisito fundamental es que se pueda administrar de manera remota, que tenga monitorización y que se auto-administre.

Una propuesta para el despliegue de servidores caché y servidores de CDNs en distintos puntos de la red. En el nivel de núcleo de red resulta costoso debido al uso de toda la red para hacer uso del contenido. Una mejor ubicación para atender las solicitudes de los usuarios móviles es el de enrutador frontera con la red móvil debido a que es un buen compromiso entre la distancia entre usuarios y el contenido. Poner los servidores de caché y CDNs en los gateways locales (un salto antes de las estaciones de radio) no es muy recomendable debido a que desde estas ubicaciones por lo general se tendrá que solicitar el contenido a otras fuentes. [13]

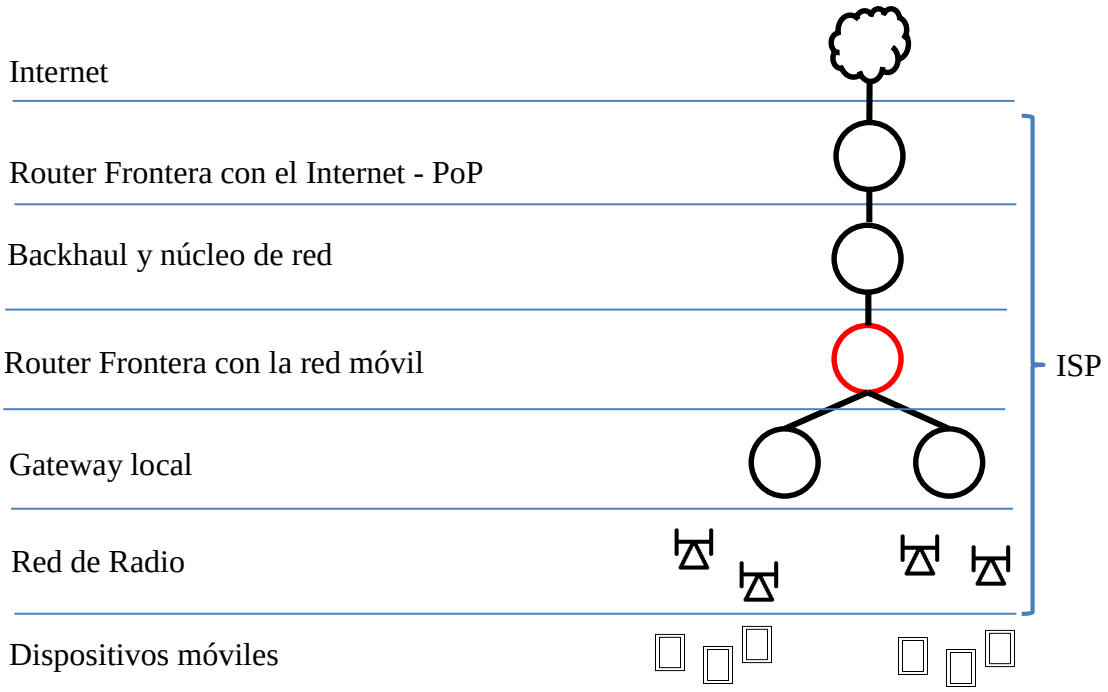


Figura 15. Distintos niveles de red

Entre las propuestas para hacer frente a las características de la red móvil se presenta DASH para entornos móviles en modos unicast y multicast para adquirir la flexibilidad frente a las redes móviles como se menciona en [56]. En la sección siguiente se mencionan los elementos principales de DASH como forma transporte de los vídeos.

## 2.4 Streaming Adaptativo Dinámico sobre HTTP (DASH)

Dynamic Adpative Streaming over HTTP (DASH) especifica formatos XML y formatos binarios que permiten la entrega de contenido multimedia haciendo uso del estándar HTTP.

El MPD (Media Presentation Description) que describe como un conjunto multimedia (Media Presentation) puede estar delimitada o no. En particular define los formatos para declarar los identificadores de los recursos y los segmentos para proveer recursos dentro del conjunto multimedia. Los recursos son URLs de HTTP que posiblemente tendrán un rango de bytes indicando el vídeo requerido.

Los formatos de los segmentos especifican las respuestas a las solicitudes HTTP GET. Los segmentos son los que contienen datos de tipo multimedia codificados. La figura 16 muestra los casos de uso del MPD y de los segmentos. Uno de los principales objetivos de DASH es tener un mismo vídeo en varias calidades. Cada calidad se segmentará en pequeños trozos. Luego el cliente irá solicitado los segmentos de una calidad y si las

condiciones de red mejoran (o empeoran) podrá solicitar segmentos de una calidad más apropiada manteniendo la continuidad de la reproducción del vídeo.

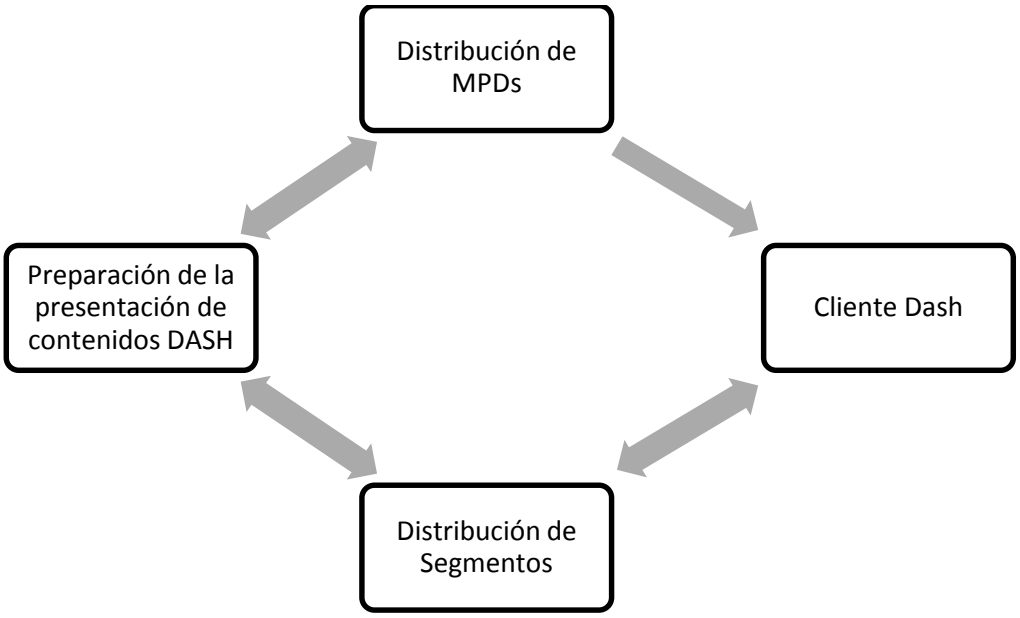


Figura 16. Sistema DASH

El MPD tienen una lista de URLs que se pueden solicitar haciendo uso de las solicitudes HTTP GET lo que permite que se utilicen de múltiples maneras. En la figura 11 podemos apreciar el modelo del cliente DASH. [17]

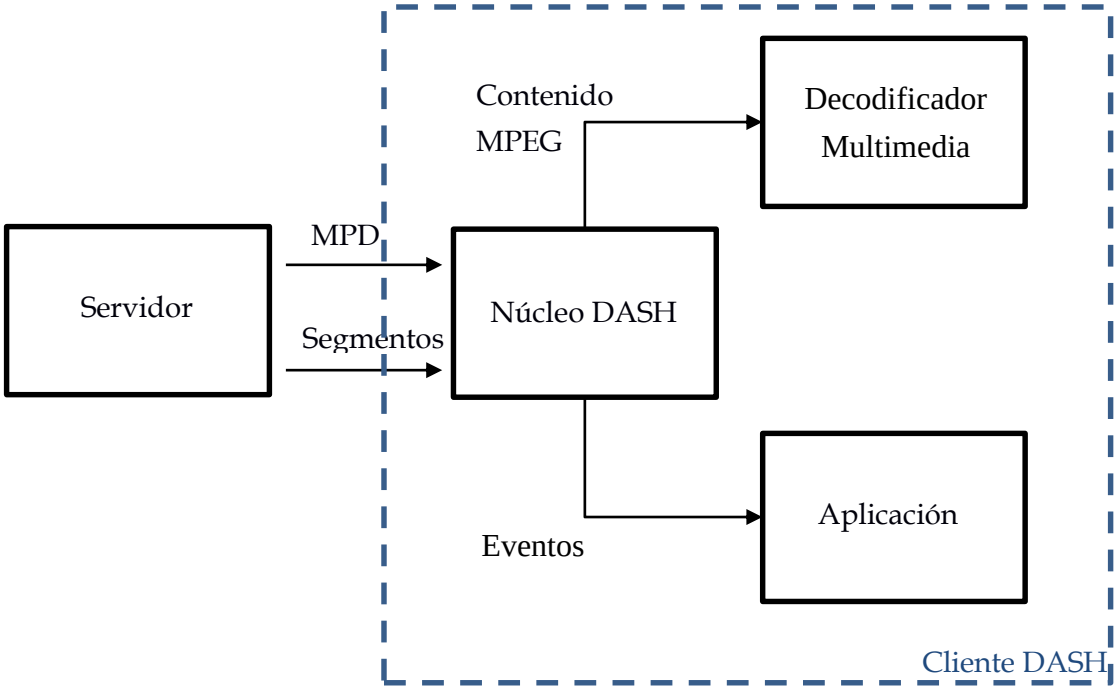


Figura 17. Modelo de un cliente DASH

El diseño de los formatos se basa en el modelo de cliente "informativo" (informative client model). La figura 18 que se presenta a continuación nos muestra la estructura de un MPD. La estructura del MPD tiene varios niveles entre los que se encuentra el período, el conjunto de adaptación, la representación, los segmentos, la subrepresentación y los subsegmentos. [53]

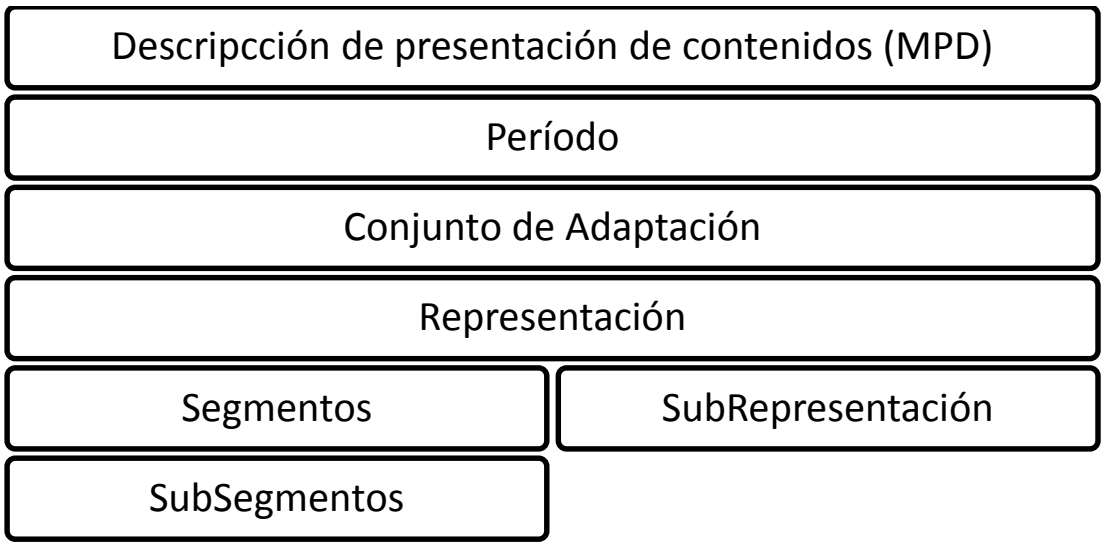


Figura 18. Estructura de un MPD

El propósito de DASH tiene por objetivo distribuir contenido multimedia haciendo uso del cliente como controlador. Los clientes pueden hacer uso del protocolo HTTP para solicitar contenido a servidores que no implemente DASH. El estándar no describe los procedimientos de los clientes o los servidores sino en los formatos de los datos usados para llevar a cabo la comunicación entre ellos y así brindar los datos multimedia DASH.

El conjunto de los ficheros multimedia codificados y los ficheros de descripción de los ficheros anteriores hace el conjunto multimedia (Media Presentation). El contenido multimedia consiste de uno o varios datos que se siguen los unos a los otros en secuencia temporal. El contenido en distintos periodos de contenido multimedia puede ser completamente independiente o algunos de los períodos pueden pertenecer a un mismo Recurso (Asset), por ejemplo el conjunto multimedia (Media Presentation) es una colección de un programa principal compuesto de muchos períodos, cada uno de los cuales está asignado al mismo Recurso (Asset), y entrelazando con periodos de propagandas (anuncios, advertisements). Cada contenido está compuesto de uno o varios componentes de contenido multimedia, por ejemplo las pistas de audios en diferentes idiomas, diferentes canales de vídeos con vistas variantes del mismo vídeo, etc. Los metadatos son importantes para realizar una selección dinámica o estática del contenido multimedia.

DASH se basa en un modelo jerárquico. El conjunto multimedia (Media Presentation) se declara mediante un documento MPD (Media Presentation Description). Esto describe una serie de Períodos ordenados temporalmente (cronológicamente) que conforman el conjunto multimedia. Un período representa contenido multimedia que está disponible por ejemplo las tasas de bits, idiomas, subtítulos que no cambian durante el Periodo.

Dentro de un período el material se ordena en conjuntos de adaptación (Adaptation sets). Los conjuntos de adaptación representan versiones intercambiables de contenido multimedia. Por ejemplo puede haber un conjunto de adaptación para el componente principal de vídeo y otro set de adaptación para el audio. Si hay otro material disponible como descripciones de audio ("música", "sonido fuerte", etc.) este material disponible puede ir en su proprio conjunto de adaptación. El material también se puede proveer en forma multiplexada, en cuyo caso distintas versiones de la información multiplexada podrán describirse como un solo conjunto de adaptación, un ejemplo de este caso sería un solo conjunto de adaptación que contenga audio y vídeo para un período. [53]

Un conjunto de adaptación tiene un grupo de representaciones. Una representación describe una versión codificada y distribuible de uno varios componentes multimedia. Una representación incluye uno o más streams de media. Cada representación dentro de un conjunto de adaptación es suficiente para reproducir el contenido multimedia. Al juntar diferentes representaciones en un solo conjunto de adaptación, el autor del MPD nos querrá decir que las representaciones presentan contenido que aparentemente es el mismo es decir que los sentidos (de la vista y oído) no podrán diferenciarlos. Esto da la posibilidad a los clientes de cambiar dinámicamente de una representación a otra dentro de un conjunto de adaptación El cambio se refiere a la presentación de la información decodificada hasta un cierto tiempo "t" para luego pasar a otra información decodificada de otra representación a partir del tiempo "t" (el mismo tiempo mencionado anteriormente). Si el cliente realiza el cambio entre representaciones la presentación del contenido multimedia no sufrirá de cortes por el cambio y el cambio tampoco se percibirá. Los clientes (reproductores) podrán ignorar las representaciones que hacen uso de códecs u otras tecnologías de reproducción que no estén soportados o no son apropiadas.

Dentro de una representación, el contenido puede ser divido en segmentos para una distribución y accesibilidad apropiadas. Para poder acceder a los segmentos, una URL se provee para cada segmento, en consecuencia un Segmento es la unidad de datos que se puede recuperar con una sola solicitud HTTP. [17]

Cabe mencionar que lo anterior no es estricto debido a que en una URL se puede solicitar un rango que indica que el Segmento está contenido en el rango proporcionado

(contenido dentro de una fuente más grande). Un cliente inteligente podría en principio construir una sola petición para múltiples Segmentos pero este no es el caso habitual.

DASH define diferentes líneas temporales. Una de las principales ventajas de DASH es que las distintas codificaciones del contenido multimedia comparten una misma línea de tiempo. La presentación de cada tiempo de cada unidad de acceso dentro del contenido multimedia se relaciona de manera global con la línea de tiempo global para la sincronización de los distintos tipos de componentes multimedia. Esto permite realizar cambios entre las diversas codificaciones de manera imperceptibles. La línea de tiempo global se llama "línea de tiempo de la presentación del contenido multimedia". Los segmentos multimedia tienen información precisa de tiempo lo que permite la sincronización y cambios imperceptibles.

Una segunda línea de tiempo es usada en los clientes para indicar a los clientes los tiempos en los cuales los Segmentos están disponibles en las HTTP-URLs. Estos tiempos son llamados Tiempo de disponibilidad de los Segmentos y se proveen en tiempo de reloj de pared (wall-clock time). Los clientes por lo general comparan el tiempo de reloj de pared (wall-clock time) antes de acceder a los segmentos en las URLs disponibles para evitar recibir respuestas HTTP erróneas. Para presentaciones de multimedia estáticas la disponibilidad de los tiempos de disponibilidad es idéntica para todos los segmentos. Para presentaciones multimedia dinámicas los tiempos de disponibilidad dependen de la ubicación del segmento en la Línea de tiempo global. Los segmentos se vuelven disponibles a lo largo del tiempo. Las presentaciones multimedia estáticas son apropiadas para ofrecer servicios "bajo-demanda" (on-demand) mientras que las presentaciones multimedia dinámicas son preferibles para ofrecer servicios en vivo y en directo (live services).

Los segmentos tienen una duración asignada que es la duración del contenido multimedia que se encuentra en el segmento cuando se reproduce a velocidad normal. Usualmente todos los segmentos en una representación tienen aproximadamente una duración similar. Sin embargo la duración de los segmentos puede variar entre representaciones. Una implementación DASH puede construirse a partir de segmentos cortos (un par de segundos), o largos también se puede incluir un solo segmento para toda la representación.

Segmentos más cortos son usados cuando se realiza la reproducción de contenido en vivo (en directo) lo que ocasiona restricciones de latencia entre el cliente y el servidor. Usualmente la duración del Segmento es menor a la latencia de la conexión. DASH no da la opción a extender los Segmentos a lo largo del tiempo debido a que los segmentos son tratados como unidades discretas y finitas que debe estar disponible de manera completa. Sin embargo, esto no impide la aplicación de modos de transmisión HTTP

avanzadas, modos como transferencia de "chunks" que permiten optimizar el despliegue y reducir la latencia entre los extremos de la conexión.

Los segmentos podrán dividirse en subsegmentos que contienen un número completo de unidades de acceso. También puede haber restricciones específicas a los formatos a los límites de los subsegmentos, por ejemplo en el formato ISO un subsegmento debe contener un número de fragmentos de vídeo. Si un segmento se divide en subsegmentos estos deberán describirse por un índice de segmentos que provee el tiempo de reproducción en la representación y su rango de bytes que ocupa dentro del segmento. Los clientes podrán descargar este índice previamente y así podrán solicitar subsegmentos específicos. [17]

Los clientes podrán cambiar de representación en representación dentro de un conjunto de adaptación en cualquier momento. Sin embargo, la realización de cambios en cualquier momento puede ser compleja debido a la codificación (y dependencias de la codificación) dentro de la representación y otros factores, que posiblemente requieran de procesamiento y descargas en paralelo por parte del cliente DASH. También se recomienda descarga datos repetidos para un mismo periodo de tiempo de múltiples representaciones. Los cambios se realizan de manera más fácil en los SAP (Stream Access Point) dentro de un nuevo flujo de datos (stream). Para poder formalizar los requisitos relacionados con los cambios dinámicos DASH define los SAP. Los SAP son puntos de acceso al flujo de datos que son independientes de los codecs.

La segmentación y subsegmentación podrán realizarse de maneras que posibiliten los cambios dinámicos de manera más simple. Un ejemplo muy simple es que cada Segmento (o Subsegmento) comience con un SAP y que los límites de los Segmentos (o Subsegmentos) estén alineados con las representaciones de un conjunto de Adaptación. En este caso los cambios dinámicos de representaciones consisten en reproducir todo el segmento (o subsegmento) de una representación y luego comenzar a reproducir el nuevo segmento de la nueva representación. El MPD (Media Presentation Description) (descripción de la presentación multimedia) y el índice de segmentos brindan información sobre las propiedades que hacen que los cambios dinámicos sean más simples. Otros documentos pueden describir de forma específica dichas características lo que puede ocasionar restricciones a cambio de tener implementaciones de clientes más fáciles. [53]

Para servicios bajo demanda el MPD es un documento estático que describe varios aspectos de la presentación multimedia. Todos los segmentos de una presentación multimedia están disponibles en el servidor una vez que cualquier segmento está disponible. Para los servicios en vivo, sin embargo los segmentos se van volviendo disponibles a medida que el tiempo va avanzando y el contenido se va generando en

este caso los MPD dinámicos son preferibles. El MPD puede actualizarse de manera periódica para reflejar los cambios realizados en los contenidos multimedia. Por ejemplo se pueden agregar nuevas URLs de los segmentos y borrar URLs que ya no estén disponibles. Sin embargo si los segmentos se describen usando un "template" puede darse el caso que las actualizaciones solo sean necesarias para tener redundancia o evitar fallos.

Se definen "eventos" que se pueden implementar en el MPD o en la Representación para representar información de naturaleza aperiódica al cliente DASH. Los eventos tendrán tiempo por ejemplo cada evento empezará en un tiempo específico del tiempo de reproducción global y usualmente tendrá una duración. Entre los eventos se encuentran eventos específicos de DASH de señalización y particulares de la aplicación. Ejemplos de eventos son las notificaciones de actualización (que tendrán como contenido del mensaje la actualización bien definida). El mecanismo de evento también puede utilizarse para distribuir información de tiempo relacionado con otros eventos de la aplicación como el momento oportuno para mostrar propagandas.

### 2.4.1 Protocolos

DASH hace uso de un cliente especificado en el RFC 2616 que ya ha sido reemplazado por [29]. El servidor HTTP que brinda los segmentos DASH cumple con lo especificado en el RFC 2616. Los clientes DASH hacen uso del método HTTP GET o de un HTTP GET parcial como se menciona en el RFC 2616 para poder acceder a los Segmentos (u otras partes).

El uso del protocolo HTTP nos permite hacer uso de técnicas modernas como el caching, la redirección o la autenticación. Como ejemplo adicional se puede hacer uso de los protocolos de seguridad como TLS para brindar seguridad a las comunicaciones en HTTP (HTTP over TLS como se menciona en [55]). Como ejemplo final de las ventajas de usar HTTP se menciona el uso de "cookies" que es el mecanismo de manejo de estado en HTTP. [54]

Un aspecto importante es que los formatos definidos en la especificación ISO/IEC 23009 pueden ser utilizados con otros protocolos. Cualquier objeto puede ser transportado por un protocolo que brinde una relación entre HTTP-URL y el objeto.

### 2.4.2 Descripción de Presentación de Medios (MPD)

El MPD es un documento que contiene la metadata necesaria para que el cliente DASH puede generar las solicitudes HTTP-URL. Una vez generados los HTTP-URL se podrá acceder a los Segmentos y así brindar el servicio de streaming al usuario. El MPD es un documento XML. [53]

### 2.4.3 Procesado de Bases URLs

Esquema XML de BaseURL

```
<!-- Base URL -->
<xs:complexType name="BaseURLType">
    <xs:simpleContent>
        <xs:extension base="xs:anyURI">
            <xs:attribute name="serviceLocation" type="xs:string"/>
            <xs:attribute name="byteRange" type="xs:string"/>
            <xs:attribute name="availabilityTimeOffset" type="xs:double" />
            <xs:attribute name="availabilityTimeComplete"
type="xs:boolean" />
            <xs:anyAttribute namespace="other" processContents="lax"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

En un documento XML se construiría de la siguiente manera:

```
:<BaseURL serviceLocation="" byteRange="" availabilityTimeOffset=""
availabilityTimeComplete="" anyAttribute= " "></BaseURL>
```

El campo BaseURL se puede utilizar en los niveles de:

- MPD
- Período
- Conjunto de Adaptación
- Representación

Si se usan varios campos BaseURL se podrá disponer de Segmentos en múltiples ubicaciones. Si no hay un criterio para utilizar los campos BaseURL, el cliente DASH utilizará el primero.

## 2.5 Resumen

En este capítulo hemos visto la base teórica, que utilizan los métodos de optimización de tráfico en la red y desarrollo de escenarios en redes de distribución de contenido, que alcanzan a los conceptos de streaming de contenidos entre los cuales se encuentran los tipos push y pull. El streaming push consiste en que el servidor envié paquetes al cliente

hasta que este termine la sesión. Los protocolos más usados en streaming tipo push son RTSP, RTP y UDP. El streaming pull consiste en que el cliente solicite el contenido que necesita. Entre las tecnologías de streaming pull se tiene la de Microsoft Smooth Streaming y la de Apple HTTP Live Streaming. Adicionalmente se ha visto los requisitos mínimos que debe tener un cliente para poder realizar el streaming adaptativo tipo pull que es la de realizar consultas HTTP GET y de utilizar TCP como medio de transporte. También se ha mencionado la tecnología SVC que permite enviar mejores calidades de vídeo enviando información adicional a la ya enviada por lo que es un sistema escalable aunque su implementación es compleja. También se han visto los casos en los cuales es más común realizar un streaming de contenidos como es el streaming desde dispositivos móviles en sitios públicos y dentro del hogar, las redes privadas de los ISP también se ha comentado sobre el uso de servidores domésticos y el uso de redes de pares (P2P).

En IPTV se ofrecen más servicios que solo el de contenidos multimedia sino que también servicios interactivos y complejos como tele-enseñanza y tele-medicina. Los terminales de IPTV tendrán capacidad para desplegar aplicaciones en HTML y JavaScript entre otros lenguajes de marcado. IPTV muestra otra filosofía de distribución de contenidos donde el contenido se encuentra dentro de ISP lo que le permite un mayor control de su red. Sin embargo las tecnologías como DASH también se pueden aplicar dentro de este contexto para la distribución del contenido de vídeo.

Las redes de distribución de contenidos son una respuesta moderna al consumo masivo de contenidos multimedia. Las redes de distribución de contenidos pueden tener distintos tipos de arquitecturas entre las que se encuentran las altamente distribuidas y las que se concentran en centros de datos. Los algoritmos de selección de las CDNs y los parámetros a tener en cuenta todavía son tema de investigación.

Se ha explicado el estándar DASH y su funcionamiento básico el cual consiste en tener un mismo vídeo en varias calidades y segmentarlos para poder cambiar de calidad a nivel de segmento en caso de que las condiciones de red cambien. El estándar DASH es una iniciativa del 2010 por lo que todavía está en desarrollo sin embargo conceptualmente y en la práctica se ha vuelto importante.

# 3 Metodología de Selección de Redes de Distribución de Contenidos

En este capítulo se realizará un análisis detallado de la selección de redes de distribución de contenidos de los proveedores. Se revisará con mayor énfasis la metodología, arquitectura, servicio, ficheros de manifiesto y medidas de desempeño utilizadas por el principal proveedor de contenidos Netflix. Asimismo, se efectuará una revisión de otras tecnologías y métodos desarrollados por los proveedores Youtube y Hulu.

## 3.1 Arquitecturas de Proveedores de Contenidos

Netflix es una de las empresas de vídeo streaming más importantes en Internet. Netflix puede realizar streaming de video de alta calidad (HD) con un bitrate promedio de 3.6 Mbps. De hecho Netflix da servicio al 29.7% del tráfico streaming en bajada. Las decisiones de manejo de tráfico y diseño de Netflix tienen un gran impacto en la infraestructura de red. El diseño de una plataforma tan grande con buena disponibilidad y escalabilidad es un reto técnico importante. La mayoría de los servidores de Netflix estaban alojados en los centros de datos privados de Netflix. Recientemente Netflix ha decidido usar servicios en cloud (en la nube) como CDNs y otros servicios públicos. Amazon AWS cloud reemplazó al centro IT de Netflix, y Amazon simple DB, S3 y Cassandra son utilizados para el almacenamiento de ficheros. El streaming de video se sirve a través de múltiples CDNs y usan a UltraDNS como servicio público de DNS. Como tecnología de reproducción de vídeos usan Microsoft Silverlight. Como resultado Netflix logra construir su sistema de distribución de vídeo con muy poca infraestructura propia. [19]

Netflix está diseñado para proveer grandes cantidades de contenidos a través del uso de servicios de terceros. Esta arquitectura se puede considerar como el plano o modelo para desarrollar proveedores de contenido sin infraestructura propia. Se comentará sobre la interacción de los componentes de la arquitectura teniendo en cuenta las múltiples CDN y el streaming adaptativo HTTP. Se analizarán los algoritmos que utiliza Netflix para coordinar y armonizar los distintos servicios de los cuales hace uso. Se verá el impacto de las decisiones de Netflix sobre la infraestructura de las CDNs, la red y la experiencia de usuario, de esta forma podremos comprender su desempeño y mejorar su diseño. Adicionalmente basado en los resultados publicados del artículo Adhikari, V.K et.al., sugiere nuevas estrategias para la entrega de vídeo que mejorarán la experiencia de usuario y usarán efectivamente las CDNs.

## 3.2 Arquitectura de Netflix

Para poder analizar la estructura de Netflix han creado una cuenta de Netflix para acceder a la página web de Netflix y reproducir una película. Durante esos procesos se ha estado monitorizando el tráfico y han almacenado los nombres de los servidores. Por medio de resoluciones de DNS han recolectado los nombres canónicos (Canonical Names CNAMES) y direcciones IP de todos los servidores que el navegador ha utilizado. También realizan búsquedas WHOIS para encontrar las direcciones IP de los diversos servidores. La arquitectura de Netflix se puede ver en la figura 19.

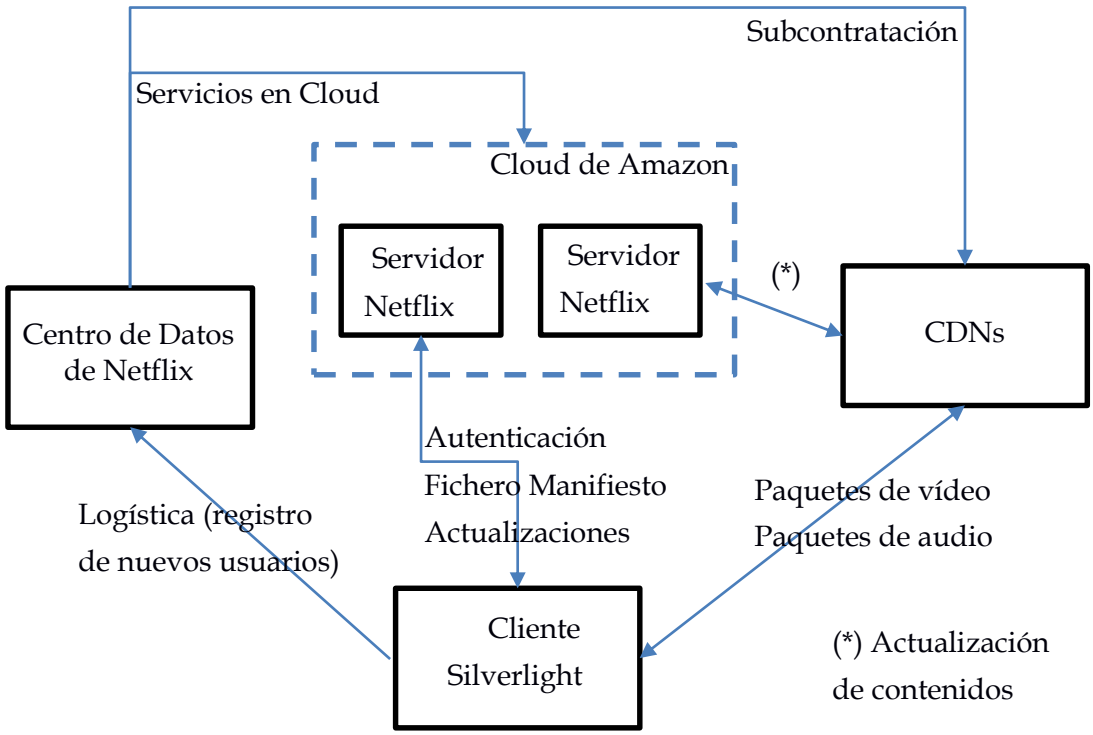


Figura 19. Arquitectura de Netflix

### 3.2.1 Centro de Datos de Netflix

Netflix hace uso de su propio espacio de direcciones IP para "www.netflix.com". Este servidor principalmente maneja dos funciones:

a) El registro de nuevas cuentas de usuario y la información de pago.
b) La redirección de los usuarios a "movies.netflix.com" o "signup.netflix.com" (dependiendo si el usuario se ha autenticado o no). Este servidor no se relaciona con el cliente durante la reproducción de la película.

### 3.2.2 Cloud de Amazon y CDNs

Es necesario mencionar que solo "www.netflix.com" está alojado en Netflix mientras que casi la totalidad de los otros servicios como "agmoviecontrol.netflix.com" y

"movies.netflix.com" son operados desde el cloud de Amazon. Esto indica que Netflix hace uso de varios servicios de Amazon como EC2, S3, SDB y VPC. Operaciones claves como la adquisición de contenido, mantenimiento de los registros (para el análisis propio de Netflix), backup, DRM (Digital Rights Management, administración de derechos digitales anti piratería), el inicio de sesión con las CDN (Redes de distribución de contenido) y el soporte para dispositivos móviles se dan en la nube de Amazon.

### 3.2.3 Redes de Distribución de Contenidos

Netflix hace uso de varias CDN para distribuir el contenido a sus usuarios. Los vídeos codificados con DRM se procesan en la Cloud de Amazon antes de ser enviadas a las CDN. Las CDNs de Netflix tienen todos los vídeos sin importar la calidad o el título del vídeo. Netflix hace uso de 3 CDNs:

- Akamai
- LimeLight
- Level-3

### 3.2.4 Reproductores

Netflix hace uso de la tecnología Microsoft Silverlight para descargar, decodificar y reproducir vídeos en los navegadores de los ordenadores. El entorno de ejecución de Silverlight está disponible para la mayoría de navegadores web. También hay reproductores para dispositivos como Wii y Roku entre otros. Aunque la forma más común de acceder a neteflix es a través del reproductor Silverlight desde un ordenador de escritorio haciendo uso de un navegador web. Netflix hace uso del protocolo DASH (Dynamic Streaming over HTTP) para el streaming. En DASH cada vídeo se codifica en diferentes calidades y se divide en "chunks" (trozos de vídeo de no más de un par de segundos). Los clientes solicitan un "chuck" o segmento de vídeo por HTTP. Con cada descarga se mide el ancho de banda y se ejecuta un "algoritmo de muestreo de tasa de descarga" para determinar la calidad del siguiente "chuck" segmento a solicitar. DASH permite a los reproductores cambiar libremente de calidad, a nivel de segmentos, dependiendo de las condiciones del canal de comunicaciones. [19]

## 3.3 Servicio a un Cliente de Netflix

Ahora prestaremos atención a la interacción entre el navegador web del cliente y varios de los servidores involucrados en el proceso de reproducción de vídeo. En la figura 20 se puede ver la interrelación de todos los elementos.

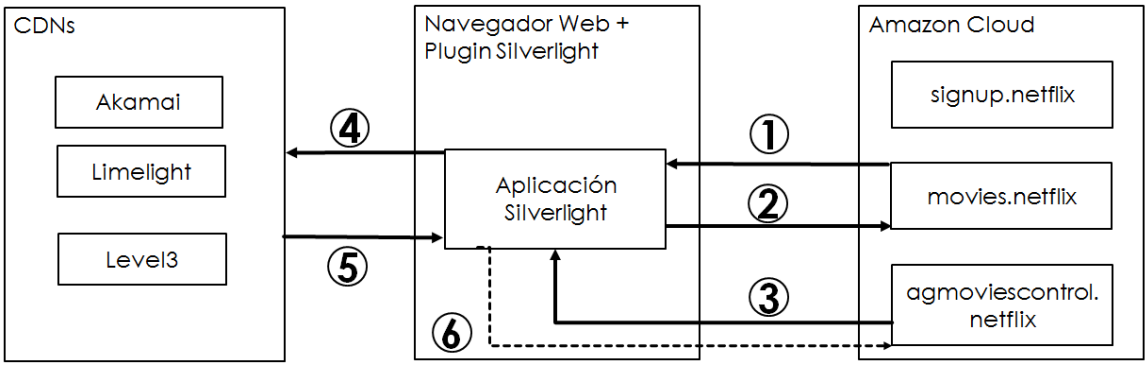


Figura 20. Procesos de Netflix

1. Primero el cliente descarga la aplicación Microsoft Silverlight de "movies.netflix.com"
2. El usuario se autentica por medio de la aplicación.
3. El reproductor obtiene el archivo de manifiesto del servidor de control "agmoviecontrol.netflix"
4. El reproductor usa el manifiesto para descargar datos "trickplay" y "chunks" segmentos de vídeo/audio de los distintos CDNs.
5. La CDN responde con los trozos ("chunks") de vídeo (DASH)
6. La aplicación de Microsoft Silverlight genera reportes que son enviados al centro de control de manera periódica.

### 3.3.1 Aplicación Microsoft Silverlight

Se requiere el plugin de Microsoft Silverlight para el navegador web. Cuando el usuario hace uso del botón de "Play Now" el navegador descarga la aplicación hecha en Silverlight y la aplicación empieza a descargar el vídeo. La pequeña aplicación de Silverlight es descargada para cada reproducción del vídeo.

### 3.3.2 Fichero de Manifiesto Netflix

El streaming de Netflix es controlado por las instrucciones en el fichero de manifiesto que el cliente de Silverlight descarga. El manifiesto provee de información de tipo meta-data al reproductor DASH para poder realizar la reproducción (streaming) del vídeo en forma adaptativa. El fichero de manifiesto es específico al cliente por ejemplo indica si el usuario puede reproducir la codificación H.264 o si puede reproducir archivos con la extensión "wmv".

El manifiesto es distribuido al cliente a través de una conexión SSL por lo que no se puede leer usando "tcpdump" o "wireshark" directamente. Para poder leer los ficheros se utilizó el plugin de "Tamper Data" para el navegador web Firefox con el propósito de extraer la información de los ficheros de manifiesto. El fichero extraído estaba en formato XML y contiene piezas claves de información como la lista de CDN, ubicación

de la información "trickplay", los URL para las distintas calidades de los vídeos, parámetros de tiempo como el intervalo "time out" y el intervalo de muestreo. En el fichero de manifiesto muestra también información interesante sobre la arquitectura de Netflix como por ejemplo el uso de 3 CDNs y sus prioridades de cara a Netflix.

### 3.3.3 Trickplay

El "trickplay" provee las funcionalidades de pausa, retroceso, adelanto y búsqueda aleatoria. Se obtiene la funcionalidad de búsqueda aleatoria al descargar imágenes para los distintos períodos de tiempo. La resolución de las imágenes, el intervalo del trickplay y la CDN preferida se indican en el manifiesto. El período de trickplay para ordenadores de escritorio es de 10s y se tienen una variedad de resoluciones dependiendo del usuario.

### 3.3.4 Descarga de Segmentos de Audio y Vídeo

Como se muestra en la figura 21 el audio y el vídeo se descargan por segmentos/trozos. Las sesiones de descarga son más seguidas al comienzo debido a que es necesario llenar el buffer. Una vez que el buffer está suficientemente lleno, las descargas se vuelven periódicas. El intervalo entre dos descargas consecutivas es de 4 segundos (lo que demora el "chunk" en reproducirse). El manifiesto también contiene información de las distintas calidades, y para cada calidad tiene un URL para cada CDN.

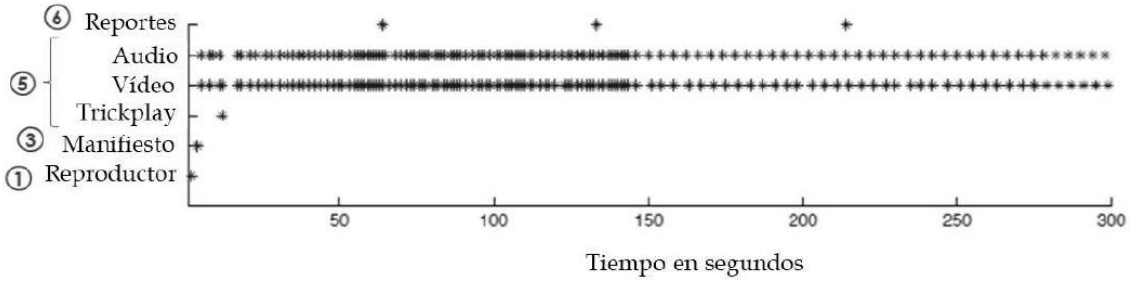


Figura 21. Procesos de Netflix en el tiempo

### 3.3.5 Reportes de Experiencia de Usuario

Luego de que la reproducción empieza, el reproductor de Netflix comunica periódicamente con el servidor de control "agmoviecontrol.netflix.com". Utiliza las palabras claves "/heartbeat" y "/logblob" para solicitar URLs y por la periodicidad de los mensajes se especula que son de tipo "keep alive" No se ha podido decodificar los ficheros usando "Tamper Data" y no se sabe que contienen (pues están en texto cifrado).

## 3.4 Análisis de los Ficheros de Manifiesto

Los ficheros de manifiesto se entregan a través de una conexión SSL. Se usó Tamper Data de Firefox para leerlos. El fichero de manifiesto contiene una gran cantidad de información (y por lo tanto la información podría cambiar de un fichero de manifiesto a otro) por lo que se procedió a reunir una gran cantidad de ficheros de manifiesto para poder realizar un buen análisis. Se ha desarrollado un experimento práctico con el objetivo de entender como la ubicación geográfica, las capacidades del cliente y el tipo de contenido (popular y de nicho, películas y series de TV) tienen un impacto sobre los parámetros de streaming del fichero de manifiesto. Se usaron 6 cuentas de usuario, 25 películas con distinto grado de popularidad, grupo de edad, género, 4 ordenadores con sistemas operativos Windows y Mac en 4 ubicaciones diferentes. Para cada ordenador, se utilizó cada una de las cuentas de usuario y se reprodujo la película por un par de minutos para obtener los archivos de manifiesto. Para obtener información adicional se usaron servidores proxy Squid que estaban alojados en 10 nodos de PlanetLab alojadas en universidades. [19]

### 3.4.1 Ranking de CDN y Cuentas de Usuario

Netflix usa los manifiestos para indicar que CDN prefiere. El ranking define la CDN que el cliente utilizará para descargar el vídeo y puede afectar la experiencia de la calidad del servicio. Se estudiaron los manifiestos para ver qué factores influyen en la prioridad de las CDNs. Para este análisis, se observó la prioridad de la CDN por cada tipo de cuenta de usuario, ubicación del ordenador del cliente (PlanetLab proxy), modelo de ordenador (MAC y ordenador convencional), tipo de película y hora del día. El análisis de esta tabla sugiere que las prioridades de las CDNs solo dependen de la cuenta del usuario. Para una cuenta de usuario determinada, la prioridad de la CDN en el manifiesto se mantiene igual sin importar la ubicación, el tipo de película o la hora del día, computadora y para dos usuarios distintos a la misma hora obtuvieron listas de prioridades distintas. También se observa que la lista de prioridades de las CDNs no cambia por un par de días. Al parecer las prioridades de las CDNs solo dependen de la cuenta de usuario y se reparten de manera aparentemente aleatoria.

### 3.4.2 Bitrates de Audio y Vídeo

Netflix provee vídeos en muchos formatos y calidades. Cuando un cliente Netflix solicita un archivo de manifiesto de Netflix, el cliente indica que formatos puede reproducir. El servidor de Netflix responde con un manifiesto basado en la solicitud del cliente. Ejemplo: un cliente que tiene un ordenador antiguo (como Thinkpad T60 con Windows XP) recibirá calidades, tasas de transferencias y formatos distintos a un ordenador moderno (MacBook Pro con Snow Leopard). Basado en las características del

cliente, el servidor manda la URL de los "chunks" de los segmentos de vídeo en el fichero de manifiesto. En general se encontraron manifiestos con segmentos codificados en tasas de transferencia entre 100Kbps y 1750Kbps (2350Kbps a 3600Kbps para HD) para los manifiestos enviados a la computadora moderna. Los contenidos en HD se pueden distribuir en 14 tasas de transferencias distintas mientras que el contenido que no está en HD solo puede distribuirlo en 12 tasas distintas. Se observó que los clientes de Netflix no intentan utilizar todas las posibles tasas de transferencia a la hora de determinar la tasa de reproducción óptima.

**Estrategia de Selección de CDN**

Se ha visto que los clientes Netflix pueden utilizar diferentes tasas de transferencia. Se realizaron experimentos para comprender como Netflix toma decisiones cuando el ancho de banda es dinámico. [19] Para esto se reprodujo una película desde el principio. Luego se utilizó la aplicación Dummynet para regular el ancho de banda entrante al cliente. Al inicio a los servidores de cada CDN se les permitía enviar datos 3900Kbps, después de cada minuto se redujo 100Kbps del ancho de banda hasta llegar a 100Kbps. En ese punto cuando ya no se puede reproducir la mínima calidad disponible Netflix pasa al siguiente CDN con lo que realiza el mismo procedimiento hasta haber utilizado los tres CDN. Netflix trata de mantener la conexión con el CDN a pesar de que otras CDN puedan brindar mejor servicio, solo realiza el cambio de CDN cuando ya no se puede reproducir el vídeo por que el ancho de banda es demasiado pequeño aún para la peor calidad.

En la figura 22 se puede apreciar como el ancho de banda se ha ido reduciendo hasta producir un cambio de CDN.

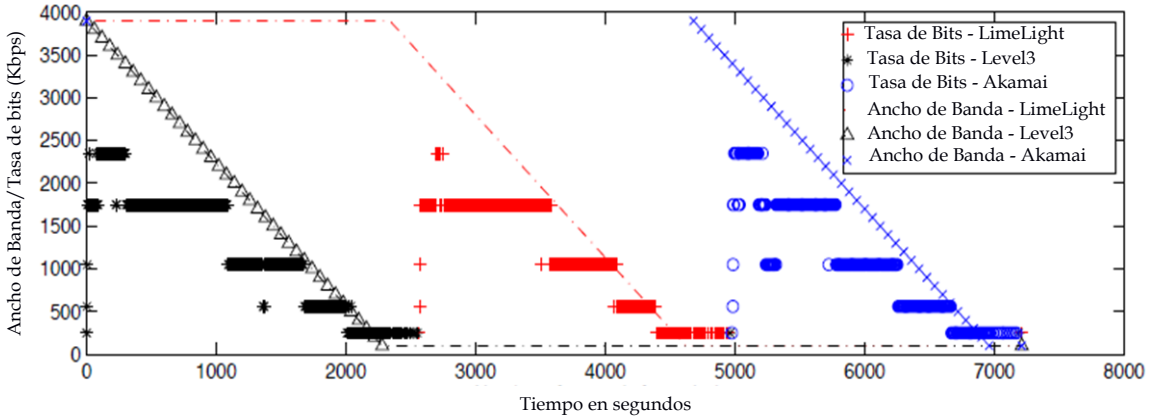


Figura 22. Comportamiento de Netflix cuando se reduce el ancho de banda

## 3.5 Medidas de Desempeño de las CDN

Para medir el desempeño de las CDNs de las que hace uso Netflix se realizaron experimentos desde 95 ubicaciones a lo largo de los Estados Unidos. Para medir el ancho de banda desde cada ubicación y una CDN se descargó varios "chunks" segmentos de vídeo desde el servidor CDN. Se consiguen varios URL de vídeos para las 3 CDN de los distintos archivos de manifiesto. En este experimento se aprovecha el hecho de que las URL en el manifiesto tienen validez por varias horas desde el momento en el cual el manifiesto es generado y la validez del URL no depende de la IP del cliente. Adicionalmente el parámetro de rango de la tasa de descarga se puede ajustar sin afectar la validez del URL(). Una vez que extraemos los URL de las 3 CDN ejecutamos un "replay" (que es una solicitud GET) desde todas las ubicaciones con una tasa de transmisión continua (debido a que con el parámetro tasa de descarga nos aseguramos que sólo una calidad esté disponible).

De manera similar a la reproducción por Netflix , cuando se envía un solicitud GET desde otra ubicación los hostnames en las URL son traducidas por el servidor frontera asignado por la CDN. Para asegurar que las medidas entre las distintas CDNs son comparables, se hicieron solicitudes GET en round bobin. Las medidas se realizaron en varios rondas, cada ronda duró 96 segundos que se dividió en 4 slots de 24 s, cada slot para una CDN. También se manda mensajes keep alive a cada servidor cada segundo aun cuando no se reciben datos para mantener abierta la conexión y para que la ventana TCP del servidor no se reduzca.

En los experimentos desarrollados por [19] se obtuvo tiempos de descarga con lo que se calculó el ancho de banda instantáneo, el promedio diario (el promedio de las dos horas del día), el promedio de ancho de banda total (a lo largo de todo el experimento). Con estos datos se puede evaluar el desempeño de las distintas CDN. Se realizaron experimentos desde lugares residenciales como desde los nodos de PlanetLab.

Se aprecia que el ancho de banda es mayor para las universidades y que la última milla sigue siendo el principal cuello de botella. Las CDNs tienen preeminencia en ciertas regiones pero no a lo largo de todo el territorio. Hay fluctuaciones en la capacidad de las CDNs.

## 3.6 Recomendaciones

Respecto a la asignación estática realizada por Netflix, el usuario solo suele utilizar una CDN pues las otras CDNs sirven solo de respaldo. Las CDNs tienen variaciones de ancho de banda a lo largo del día, entre día y por ubicación geográfica. Una dificultad de la asignación estática es que no se hace uso de las mejores CDN por zonas específicas y en el peor de los casos puede ser que le toque una CDN que es mala en una zona mientras que hay otra que si le puede brindar una mayor calidad. Al mejorar el ancho de banda a los usuarios seleccionando la mejor CDN regional no solo se mejora el servicio sino que se pueden brindar más servicios como 3D- TV o proveer de varias películas al mismo tiempo.

### 3.6.1 Ancho de Banda Ideal

Si se tiene obtiene el ancho de banda instantáneo de las 3 CDN la elección óptima es escoger la mejor CDN en cada segundo. Aunque es inviable en la práctica pues no se tienen medidas antes de recibir el vídeo, se propone como límite superior del ancho de banda que un usuario puede recibir de manera teórica.

Se usa el promedio de las 3 CDNs como muestra de la asignación estática, dependiendo de cuál se escoja los resultados serán mejores o peores pero el promedio nos da el valor esperado. Si escogieras la mejor CDN a cada segundo se daría una mejora de entre el 15% al 33%, pero si se escoges solo la mejor CDN solo se pierde un 6% a 7% del óptimo. Si se pudiera predecir que CDN será la mejor para las próximas 2 horas puedes llegar a estar cerca del límite ideal.

### 3.6.2 Escoger la Mejor CDN Basándose en Medidas

Para identificar la mejor CDN se pueden realizar varias medidas de ancho de banda al comienzo del proceso de reproducción y luego asignar la mejor CDN para el resto de la película. Con más de 2 medidas se mejora 12% respecto a la estrategia estática.

### 3.6.3 Uso de Varias CDNs

Si se usan varias CDNs en simultáneo se puede tener un mayor ancho de banda esto es posible debido a que el vídeo se transfiere en segmentos ("chunks"). La identificación del protocolo que pueda hacer uso efectivo de varias CDN está fuera del enfoque del artículo. Fuera de la forma de poder usar varias CDNs se calculó que usar las 3 CDNs en simultáneo hace una mejora considerable a la velocidad de streaming (en teoría). En la figura 23 se puede ver una comparación entre las 3 CDNs juntas contra la mejor CDN.

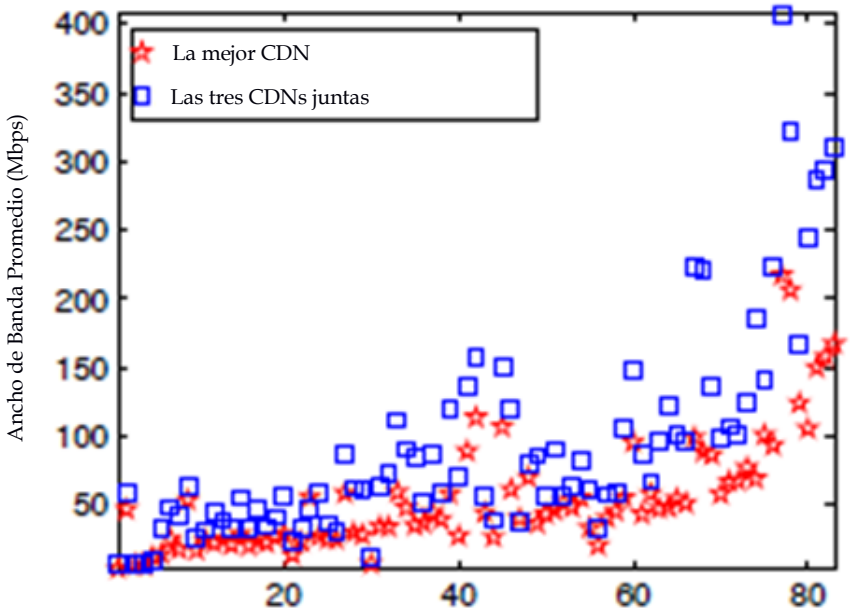


Figura 23. Comparación del uso de la mejor CDN con las tres combinadas

## 3.7 Youtube

Youtube es una referencia como fuente de vídeo en la Internet. La arquitectura de Youtube ha cambiado desde su adquisición por Google. Para el estudio se utilizaron herramientas similares como whois y se realizó análisis de tráfico desde diferentes ubicaciones. Una diferencia importante es el uso de muchos más puntos de observación y la utilización de CBG para identificar el espacio de direcciones. [20]

Se identificó las ubicaciones físicas de los servidores y se las relacionó con las lógicas. Con la ubicación física de los servidores ya se comenzó a evaluar como las solicitudes del usuario son mapeadas a YouTube. Los mecanismos son dos: uno basado en resoluciones de DNS que devuelve una dirección IP en un centro de datos. La segunda forma hace uso de funciones en la capa de aplicación para redirigir al cliente a otro servidor en otro centro de datos. La selección del centro de datos depende del origen de la solicitud del vídeo debido a que para cada sitio había centros de datos preferidos.

El comportamiento actual de Youtube es distinto al anterior registrado a la compra por Google, el comportamiento anterior de YouTube consistía en asignar las solicitudes de usuarios de acuerdo al tamaño del centro de datos. El estudio indica que el valor del RTT entre los centros de datos y el cliente juega un rol en la selección del centro de datos preferido. Hay otros parámetros como balanceo de carga, variaciones de las DNS dentro de la región, poca disponibilidad de algunos vídeos con pocas vistas y la necesidad de aliviar la carga provocada por vídeos muy populares. Los parámetros a considerar a la hora de desarrollar un algoritmo de selección de CDNs son variados.

Los pasos para ver un vídeo de Youtube se mencionan a continuación:

- En primer lugar se llega a la página del vídeo mediante un URL, solo se muestra información estática
- Como segundo paso cuando el vídeo que se desea se selecciona se va a una página web donde el vídeo esta embebido en un reproductor Flash.
- En tercer lugar el nombre del servidor que va a brindar el vídeo se encuentra en la URL. El servidor de contenidos se resuelve mediante una resolución de DNS.
- Por último el cliente solicita el vídeo al servidor de contenidos a través de HTTP.

En la figura 24 se muestran los pasos mencionados anteriormente con algunos elementos de la arquitectura de red de Youtube.

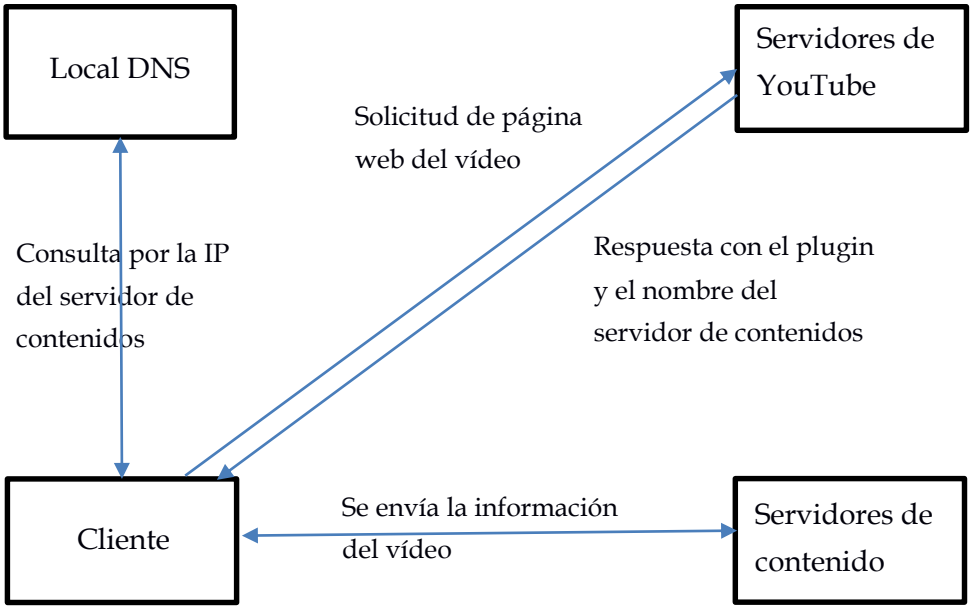


**Figura 24. Procedimiento para acceder a los vídeos de YouTube**

Se menciona que las bases de datos de Internet de tipo públicas son imprecisas para saber la ubicación geográfica de las diversas IPs. La base de datos Maxmind ubica a las bases de datos de Youtube en Mountain View (en California) sin embargo, cuando se realizan medidas desde Europa el RTT es demasiado pequeño para que este solicitando información a servidores fuera de Europa. Se encontró que YouTube tiene 13 servidores en los Estados Unidos, 14 en Europa y 6 en otras ubicaciones del mundo.

Con respecto a los algoritmos de selección de servidores se puede apreciar dos casos de uso: El primero en el cual el servidor brinda el contenido y un segundo caso de uso cuando se realiza una redirección. También para el análisis se realizó una clasificación de los flujos de datos en flujos de vídeo y flujos de control. Los flujos de vídeo consisten en conexiones largas que llevan el vídeo. Los flujos de control se caracterizan por ser conexiones cortas que llevan señalización.

Observaciones del comportamiento de Youtube:

- Hay servidores preferidos.
- El servidor con el RTT más pequeño es el que atiende la mayoría del tráfico. (85% del total). La proximidad geográfica no es el principal criterio sino el RTT (que por lo general coincide con el más cercano a los usuarios).
- Hay dos mecanismos que determinan el acceso a los otros centros de datos.
- Por resolución de DNS: Para los servidores en Europa se hizo un balanceo de carga por parte de los servidores DNS en los que se reparte la carga durante el día pero no durante la noche. Se observa para los servidores de Estados Unidos que en los casos que se usan los servidores extras (no el preferido) es debido a que el DNS que atiende dichas consultas está configurado para realizar un balanceo de carga y distribuir las solicitudes a DNS de terceros.
- Por redirección del servidor: Se observó que la mayoría de los accesos a los servidores secundarios se dan cuando hay picos de acceso a los vídeos más populares (vídeos del día). Estos procesos se realizan en la capa de aplicación. Adicionalmente ha observado que hay redirección a los centros de datos secundarios cuando se solicitan vídeos con poca popularidad.

## 3.8 Hulu

En hulu.com esta almacenada la página de Hulu donde los clientes se suscriben y selecciona los vídeos que desean ver. El servidor s.hulu.com es el encargado de enviar los manifiestos a los clientes con la información de las tasas de bits de los vídeos y los posibles servidores de donde obtenerlos. El servidor t.hulu.com se encarga de monitorización periódica. En la figura 25 se aprecian los componentes de la arquitectura de Hulu. [21]

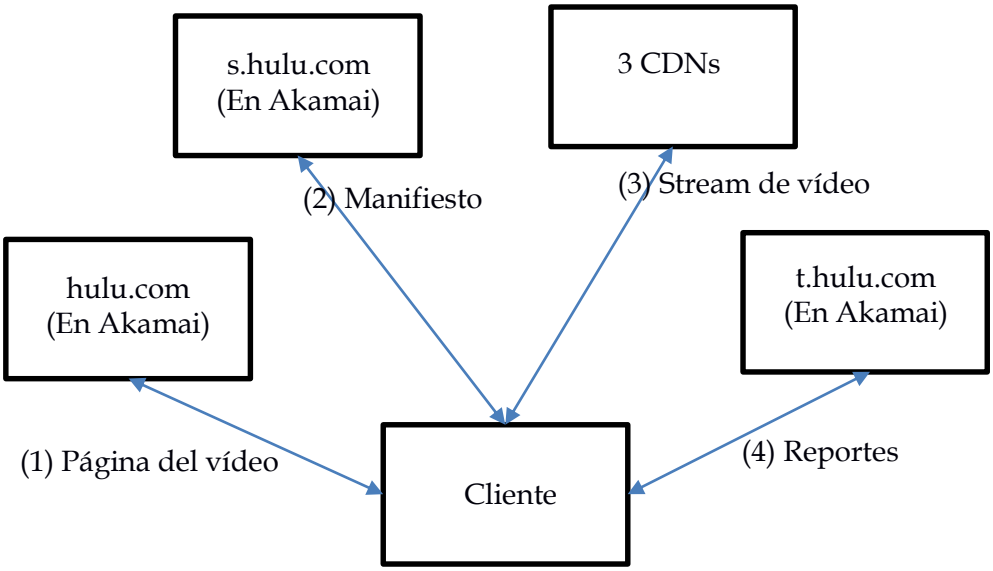


Figura 25. Arquitectura de Hulu

Los vídeos en Hulu suelen transmitirse en velocidades de 480Kbps, 700Kbps y 1000Kbps. Para la transmisión de los vídeos usa el protocolo RTMP con cifrado. Los vídeos también se pueden enviar en claro a través de RTMP haciendo uso del puerto 1935 o haciendo uso de túneles RTMP (RTMPT) sobre HTTP.

Se sabe que hay distintos factores que afectan la decisión de tomar como el ancho de banda del cliente con las CDNs, también como el desempeño pasado de las CDNs y razones no técnicas como las tarifas por el uso de las CDNs o los contratos. Del análisis de las trazas de paquetes se ha notado que Hulu envía reportes al servidor t.hulu con información detalla del estado de la máquina del cliente en ese instante, la información de los servidores CDN y cualquier fallo que haya habido en el pasado próximo. La información contenida en los reportes incluye datos de la tasa de vídeo, la posición del vídeo, la memoria total que el cliente utiliza, el ancho de banda del cliente, cuadros perdidos y el número de veces que la tasa de transmisión es menor a la velocidad de reproducción del vídeo. Cuando el cliente realiza un cambio de tasa de bit el reporte también incluye la razón por la cual se realizó el cambio en la tasa de bits. Al parecer Hulu recolecta suficiente información para realizar una selección de CDN en base a la experiencia del usuario.

Para entender la estrategia de selección de Hulu se solicitó ficheros de manifiesto para un mismo vídeo desde un solo ordenador por 100 segundos. El resultado indicó que Hulu no realiza una selección de la CDN en base a las condiciones de red sino en base a probabilidades que se mencionan a continuación: Level3 - 47%, Akamai - 28% y Limelight – 25%.

Estas preferencias no cambiaron cuando se solicitaron distintos vídeos es decir se mantuvieron independiente del vídeo seleccionado. También las preferencias se mantuvieron a lo largo de 24 días aunque se realizaron cambios de ubicación desde la cual se realizaba la solicitud. De estos resultados se propone que tal vez el criterio que explique este comportamiento sean los acuerdos comerciales entre las empresas. También se aprecia que se trata de mantener la CDN con la que se empieza y si un usuario comienza con una CDN con una calidad baja el usuario por lo general se quedará con dicha calidad de servicio por la duración del vídeo.

La selección de un servidor consiste de los pasos de seleccionar una CDN y seleccionar luego un servidor dentro de dicho CDN. Vemos como Hulu atiende a sus distintos tipos de clientes de ordenadores, móviles y su publicidad. Los tres servidores son utilizados para atender solicitudes de ordenadores mientras que solo dos son utilizados para atender a los clientes móviles y la publicidad. Se observó que los servidores de cada CDN atendía solo a un tipo de cliente, es decir si un servidor de Akamai atendía ordenadores ya no atendía a clientes móviles o publicidad mientras que otro servidor dentro de Akamai podría haber atendido a los clientes móviles pero no a los ordenadores ni publicidad.

En los experimentos realizados sobre Akamai y Level3 se descubrió que estos solo utilizaban un nombre de dominio cada uno mientras que Limelight usaba como 1000 nombres de dominios para distribuir los vídeos de Hulu a los clientes en ordenadores. Adicionalmente se observó que Limelight usa el identificador del vídeo para resolver el servidor que lo va a atender por ejemplo el vídeo con el identificador 50061220 fue atendido por hulu-220.fcod.llnwd.net. Dicho enfoque es poco flexible para realizar un balanceo de carga entre los servidores.

Para atender a los usuarios móviles que hacen uso de dispositivos iOS las CDN disponibles son Akamai y Level3 que usan un solo nombre de dominio: httpls.hulu.com para Akamai y httpls-1.hulu.com para Level3. En cambio para desplegar la publicidad de Hulu se utilizan las CDN de Akamai y Limelight.

## 3.9 Resumen

En el presente capítulo hemos visto la arquitectura de Netflix que es uno de los principales proveedores de contenidos. La arquitectura de Netflix es moderna al usar servicios en la nube como infraestructura principal. Netflix establece un modelo de arquitectura de en la red al usar los servicios web de Amazon para implementar la parte con control de sus sistema de vídeos. Adicionalmente utiliza CDNs para encargarse de la distribución de sus contenidos. También hemos observado el proceso completo que se realiza en el cliente desde que se solicita el vídeo hasta que se distribuye. Como parte este proceso hemos visto que las ubicaciones de los videos se mencionan en un fichero

de manifiesto a la aplicación en el cliente. Una observación muy importante es que para escoger la fuente del vídeo se usan listas de preferencias estáticas (que no consideran las condiciones de red) o por probabilidades. Esto es importante porque nos da una idea de cómo funciona un proveedor de vídeo y nos da pie a proponer mejoras.

Estas metodologías se podrían mejorar si es que se usarán medidas provistas por los clientes web para seleccionar la CDN con mejores prestaciones. En el caso de Youtube el parámetro principal a tener en consideración a la hora de seleccionar el servidor que brinda el vídeo es el RTT. Hay que tener en cuenta que los contratos comerciales pueden tener un impacto importante en la toma de decisiones a parte de las consideraciones técnicas. También se mencionan las tecnologías de los clientes que se utilizan para reproducir el vídeo que son: Microsoft Silverlight para Netflix, FlashPlayer para Youtube. Otro detalle es el hecho que solo se cambia de CDN cuando esta es fuera de servicio. En el caso de Hulu se menciona una selección de CDNs en base porcentajes. Sin embargo el uso que hace de sus CDNs puede que no sea el mejor al usar una selección de la fuente estática y que se basa en el usuario además la selección que le toca al usuario dura unos días sin importar si brinda un buen o mal servicio.

Entre las mejoras teóricas que se proponen se encuentran la de seleccionar la fuente de vídeo con mejores prestaciones al inicio de la reproducción y luego ir cambiando a otra con mejores condiciones si la oportunidad se presenta. Otra propuesta es la de usar varias fuentes de vídeo en simultaneo para obtener los fragmentos.

# 4 Herramientas

## 4.1 Introducción

En marzo de 2013 el grupo DASH-IF realizó una encuesta a los trece principales proveedores de contenidos en Europa. La encuesta se centró en temas relacionados con el uso del streaming adaptativo DASH. Entre los resultados de la encuesta se encontró que la mayoría (76,9%) de los despliegues de DASH se darían entre la segunda mitad del 2013 y la primera mitad del 2014 [16]. Entre las preocupaciones de los proveedores de servicio se mencionó que la falta de clientes con capacidad de reproducir el contenido DASH era la mayor preocupación, la segunda era la disponibilidad de herramientas para generar contenido DASH. La primera consideración está siendo abordada por el DASH-IF mediante la publicación libre de reproductores DASH multiplataforma. Adicionalmente esta implementado un reproductor DASH en JavaScript para navegadores web. Entre las plataformas que los proveedores buscan tener soporte se encuentran los dispositivos Android, iOs, Windows, navegadores web, smart-tvs y dispositivos de cable. Los proveedores de contenidos esperan tener servicios en vivo y bajo demanda, solo la mitad piensa desplegar contenido de pausa en directo. Con respecto al punto anterior se puede comentar que las guías de implementación DASH-AVC/264 en combinación con los perfiles DASH son apropiados para desplegar contenido libre, bajo demanda y de pausa en directo. Cuando se consultó por el retardo extremo a extremo para contenidos en vivo los proveedores expresaron su deseo de que el retardo fuera de 10 segundos o menos. Con respecto al uso de DRM (solo 12 respondieron) un 60% no está considerando utilizar un esquema DRM [16]. Sin embargo algunos utilizarán HTTPS para brindar seguridad en la capa de aplicación, adicionalmente están considerando el uso de HLS. Los proveedores demostraron su preferencia por los esquemas de DRM PlayReady (Microsoft), Marlin, Widevine y Verimatrix. Los contenidos serán en su mayoría con un solo canal de vídeo y uno o varios canales de audio seguidos de canales de texto y en menor cantidad habrá contenidos con múltiples señales de vídeo. En las proyecciones de los proveedores se comentó que había una preferencia por desplegar audio multicanal 5.1. Adicionalmente se dio a conocer que las empresas de vídeo buscaban en su mayoría reemplazar sus tecnologías de Adobe HDS, Microsoft y Apple HLS. Entre las nuevas tecnologías a ser implementadas las empresas de vídeo mostraron mayor interés en la tecnología HEVC, tecnología sobre la cual DASH está trabajando para desarrollar un DASH-HEVC de manera análoga al DASH-AVC/264, también se mostró bastante interés por mostrar una segunda pantalla [16]. Con respecto a las certificaciones más 60% de los proveedores de

vídeos consideraron la certificación funcional como necesaria mientras que el resto no lo considero necesaria. Otras certificaciones que se tomaron en cuenta fue que se consiguiera un mínimo de rendimiento, protección para que no se grabe el contenido y en menor medida que la propaganda no se pueda evitar. Entre los beneficios de DASH que los proveedores apreciaron se encuentran la estabilidad de un estándar abierto, el potencial para ser multiplataforma, facilidad de empaquetamiento y distribución, suporte DRM e inserción de propagandas. El logo del Foro de la Industria de DASH se muestra en la figura 26.


Figura 26. Logo del Foro de la Industria de DASH

## 4.2 Dashjs

Dash.js es una iniciativa para desarrollar un reproductor multimedia que pueda ser utilizado en aplicaciones comerciales. El reproductor será capaz de reproducir multimedia de tipo MPEG-DASH haciendo uso de librerías de lado de cliente de JavaScript [24]. El reproductor hace uso del Api "Media Source Extensions" definido por el W3C. Los objetivos principales son construir una librería abierta que:

- Sea agnóstica al código
- Permita múltiples tipos de encapsulamiento
- Permita múltiples esquemas de protección de contenidos
- Permita la modificación de las reglas de selección de bitrate, de los métodos de carga, de la lógica y manejo de errores.

La implementación buscará mantener la flexibilidad mencionada para poder desarrollar un reproductor que cumpla con las recomendaciones DASH264 de DASH-IF [18]. Entre las principales recomendaciones se mencionan:

Uso del estándar H.264 en vídeo y HE-ACC para el audio

Segmentos con el formato Base Media File

Soporte para los perfiles:

- urn:mpeg:dash profile:isoff-on-demand:2011
- urn:mpeg:dash profile:isoff-live2011

Alienación temporal (time aligned), uso de segmentos GOP (group of pictures) cerrados que su duración no varíe más del 25% de la media.

El streaming adaptativo HTTP nos permite brindar un streaming robusto, con acceso aleatorio que se ajusta a los cambios en ancho de banda evitando el "rebuffering" (evitar paradas hasta que el buffer vuelva a tener suficientes datos para la reproducción). El uso del puerto HTTP:80 permite el tránsito a través de toda la red, proxies y firewalls. El MPEG-DASH provee el estándar ISO/IEC 23009-1 que combina las mejores características del HLS de Apple, HDS de Adobe y el Smooth Streaming de Microsoft para producir un formato con gran flexibilidad.

El estándar ISO/IEC 23009-1 es lo suficientemente flexible para soportar varios tipos de esquemas de direcciones, soporte TS y BMFF para el encapsulamiento de segmentos, streams con varias pistas de audio o vídeo, metadatos de tiempo e inserción de propagandas. En vez de solicitar que cada navegador soporte DASH de manera directa, las extensiones de Media Source exponen una interfaz al elemento de vídeo HTTP para permitir la transmisión de segmentos de vídeo. Esto permite al cliente interpretar manifiestos, solicitar segmentos y aplicar la lógica de conmutación mientras deja la visualización de los segmentos al navegador web. El resultado es una plataforma flexible con la cual construir reproductores multimedia multiplataforma [17].

El reproductor Dashjs está organizado por el Dash Industry Forum, que es una organización sin ánimo de lucro que se estableció para fomentar la adopción del estándar MPEG-DASH. Sus miembros incluyen Microsoft, Qualcomm, Ericsson, Adobe, Sony, Cisco, Intel y Akamai entre otros.

Todo el código del reproductor Dashjs está cubierto por la licencia BSD3. Esta licencia permisiva hace posible la redistribución y uso del código y formas binarias con o sin modificación sin costos de licencia. El propósito de brindar el código de manera libre es la posibilidad de su uso para la construcción de reproductores personales, para compañías o de uso comercial.

En la figura 27 se puede apreciar los elementos principales de la aplicación que son el campo "Stream" donde se selecciona la ubicación del fichero MPD y la zona donde se visualizará el vídeo.

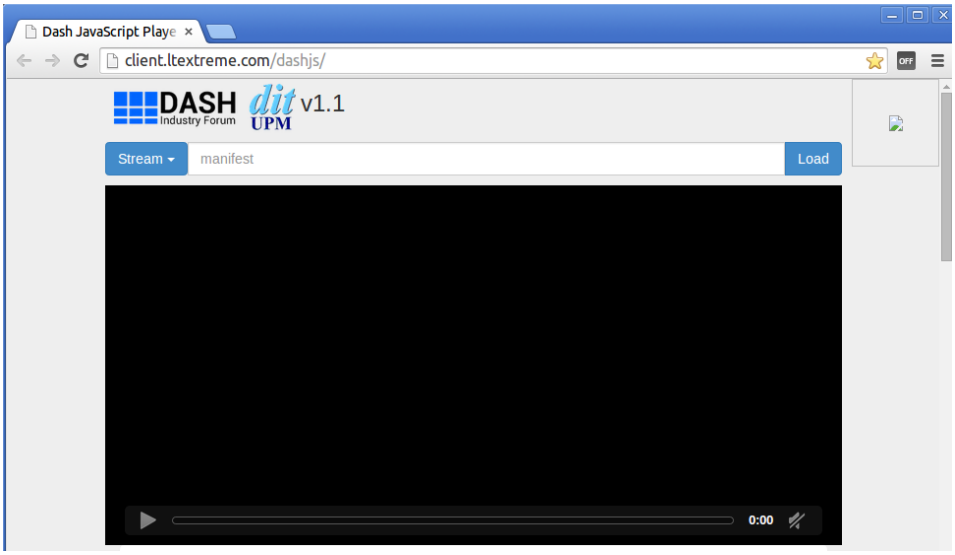


Figura 27. Captura de elementos principales de dash.js

Una vez que el usuario selecciona la fuente del MPD. La aplicación web realiza el proceso de obtención y carga del fichero. Para esto usa un objeto MediaPlayer que intentará obtener los segmentos necesarios. Para esto el objeto usará el objeto StreamController que a su vez utilizará el objeto ManifestLoader para obtener e interpretar el fichero MPD. Con la información del MPD se procederá a solicitar los segmentos y reproducirlos en la página web. Adicionalmente, en la figura 28 presentamos un esquema de funcionamiento de Dash.js.

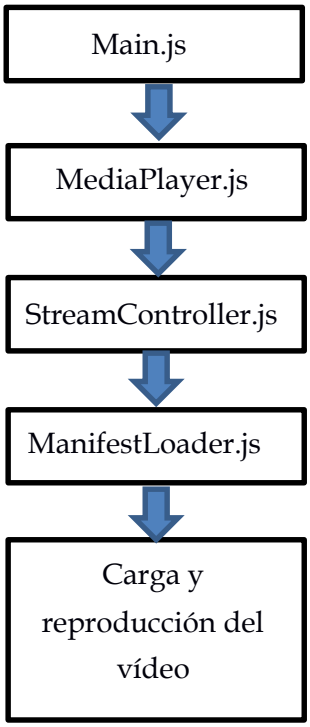


Figura 28. Carga de un MPD de un vídeo

### 4.2.1 Media Source Extensions

Cabe mencionar que la especificación de Media Source Extension es un draft del W3C por lo que todavía no es una especificación estable y puede tener cambios que afecten al reproductor Dashjs.

La especificación de Media Source Extensions permite extender el elemento HTMLMediaElement para permitir que JavaScript genere los streams a ser reproducidos. Al permitir que JavaScript genere los streams se facilitan muchos casos de uso como el streaming adaptativo y el cambio de tiempo de reproducción de los streams. La especificación permite a JavaScript construir dinámicamente streams de audio y vídeo. Con esto JavaScript puede construir los streams independiente de cómo se obtiene el contenido. Se definen modelos de buffering y particionamiento que facilitan los casos de streaming adaptativo, inserción de propagandas, edición de vídeo y cambios en la línea de tiempo. Se minimiza la necesidad de parseo de multimedia en JavaScript. Se trata de utilizar la capacidad de cacheo del navegador web. Se brindan los requisitos para el formato de los bytes. No se requiere soporte para ningún tipo particular de media o códec. Los objetos principales son el MediaSource Object, el SourceBuffer Object, el VidoPlaybackQuality Object, TrackDefault Object, TrackDefaultList Object, URL Object Extensions entre otros.

### 4.2.2 Estándar H.264

La recomendación ITU-T H.264 representa una evolución de los estándares de vídeo existentes (ITU-T H.261, ITU-T H.262 y ITU-T H.263) y fue desarrollado en respuesta a la creciente necesidad por mayor compresión de imágenes en movimiento para varias aplicaciones como videoconferencias, almacenamiento de contenidos digitales, distribución de televisión, streaming por Internet y comunicaciones. También está diseñado para permitir el uso de representaciones codificadas de vídeo de una manera flexible. La recomendación permite la manipulación de vídeo como datos de ordenador, su almacenamiento y transmisión. El estándar fue desarrollado por el ITU-T Video Coding Experts Group (VCEG) y por el ISO/IEC Moving Picture Experts Group (MPEG) que formaron un equipo llamado Joint Video Team (JVT) en el 2001 [9].

La codificación de la representación especificada en la sintaxis está diseñada para permitir una alta codificación para una calidad de imagen deseada. Por lo general el algoritmo es con pérdidas (caso de excepción de la codificación de los perfiles en High:4:4:4 Intra, CAVLC 4:4:4 Intra, High 4:4:4 predictivo, y el modo de operación I_PCM) ya que los valores exactos no se preservan a lo largo de los procesos de codificación y decodificación. Hay varias técnicas para lograr una compresión muy

eficiente. Los algoritmos de codificación pueden seleccionar entre codificaciones tipo inter o tipo intra para los bloques de cada figura. La codificación tipo Inter usa vectores de movimiento para la predicción de bloques para hacer uso de dependencias (estadísticamente importantes) entre las diversas imágenes. La codificación tipo Intra usa varios tipos de predicción espacial para hacer uso de las dependencias en la señal fuente de una sola imagen. Como se utilizan predicciones y se descarta información las imágenes si bien se aproximan muy bien a las originales no lo son.

Cabe mencionar que la especificación menciona como utilizar SVC (scalable video coding). Se especifica la construcción de streams de bits que tienen sub-stream de bits que cumplen con la especificación H.264. Para que los flujos de bits tengan escalabilidad temporal (si se tiene un sub-flujo de bits con un muestreo menor que el flujo de bits), se quitan unidades de acceso completas del flujo de bits para deducir el sub-flujo de bits. En dicho caso, la sintaxis de alto nivel y las imágenes de referencia para la predicción tipo inter-frame. En el caso de considerar la escalabilidad espacial y de la calidad (cuando hay sub-flujos con menor resolución de imagen o calidad que el flujo de bits), las unidades NAL (network abstraction layers) se quitan del flujo para generar el sub-flujo. En dicho caso, la predicción tipo inter-frame se usa para una codificación eficiente.

La codificación de vídeos con múltiples vistas también se especifica permitiendo que se utilice H.264 en flujos de bits con múltiples vista. De manera similar al SVC (scalable video coding) pueden tener sub flujos de bits que también cumplen con la especificación. Para conseguir la escalabilidad temporal, espacial y de calidad se procederá de manera análoga a la codificación de vídeo escalable SVC.

### 4.2.3 Estándar HE-ACC

Es un formato de codificación de audio digital. Es una extensión de estándar MPEG-2 ACC LC. Una de sus características principales es que esta optimizado para tener una alta tasa de bits. Es una buena opción si se quiere obtener una buena calidad de audio si se tienen bajas tasas de bits. El Perfil HE-ACC se define en la norma ISO/IEC 14496-3:2001

### 4.2.4 Subtítulos SMPTE-TT

SMPTE-TT es un estándar del SMPTE (Society of Motion Picture and Television Engineers) que especifica la implementación de la aparición de texto a lo largo de la reproducción del vídeo (Timed Text). Uno de los objetivos del estándar hacer más útil los subtítulos para que sean de igual importancia que el vídeo o el audio. El perfil que se define es el TTML (Timed Text Markup Language) del W3C. En la especificación se define los estilos sintaxis (etiquetas y jerarquía) de que se tiene a la hora de definir los subtítulos como por ejemplo el color de los subtítulos.

### 4.2.5 CENC DRM

Se refiere a la protección de medios digitales por medio del cifrado común (CENC common encryption). La norma ISO/IEC 23001-7 especifica el uso del cifrado y los mapeos de las claves que se pueden utilizar por los distintos tipos de derechos digitales. También normaliza la administración de los derechos digitales (DRM) que permiten la codificación del mismo fichero usando distintos esquemas DRM. Los esquemas operan mediante el uso de un formato común para el cifrado de los metadatos necesarios para el sistema DRM. Por ejemplo los sistemas DRM que soporten CENC deberán soportar identificar la clave de decodificación por medio del valor KID (identificador de clave) pero como se localiza la clave se deja como un procedimiento propio del DRM. La información específica DRM como las licencias o derechos se puede guardar en ficheros tipo ISO Base Media mediante el uso de pssh (Protection System Specific Header) y teniendo en cuenta el uso de uno por sistema DRM.

### 4.2.6 Interoperabilidad de Dash.js

En la versión 2.90 (13) del documento: Guidelines for Implementation DASH-IF Interoperability Points del 26 de agosto de 2014 [18].

Se espera el cliente por lo menos pueda:

- Presentar vídeo en alta definición hasta 720p (en base a H.624/AVC con el perfil Progresive High)
- Presentación de audio en estéreo
- Soporte básico para subtítulos
- Soporte básico para cifrado y DRM

Dash tiene interoperabilidad con:

Vídeo

- DASH-AVC/264, DASH-AVC/264 SD, DASH-AVC/264 HD,
- DASH-HEVC/265 DASH-HEVC/264 1080 8 bit, DASH-HEVC/265 1080 10bit

Audio:

DASH-IF audio multi-canal con:

- Enhanced AC-3
- Dolby TrueHD
- DTS Digital Surround
- DTS-HSD High Resolution
- DTS-HD Master Audio
- DTS Express

- DTS-HD Lossless
- MPEG Surround
- HE-AAC v2 nivel 4 y 6

## 4.3 Lenguaje de Programación Javascript

JavaScript es el lenguaje de programación de la Internet. La mayoría de los sitios web usan JavaScript y la mayoría de los navegadores web disponibles en ordenadores, tablets, smartphones, consolas de videojuegos incluyen un intérprete JavaScript. Es por este hecho que JavaScript es el lenguaje que está en más lugares que cualquier otro en la historia. Los desarrolladores web deben aprender por lo general tres lenguajes: HTML para la estructura, CSS para la presentación y JavaScript para especificar el comportamiento. JavaScript es un lenguaje de alto nivel, dinámico, sin tipos (no se menciona que tipo de valor es una variable como un número o un caracter) e interpretado que es apropiado para programación orientada a objetos y de tipo funcional. JavaScript usa una sintaxis basada en Java, sus funciones de Scheme y la herencia de prototipos de Self. [28]

JavaScript fue creado por Netscape. Sun Microsystems (comprado por Oracle) es el dueño de la licencia de JavaScript implementada por Netscape (en la actualidad es Mozilla). Netscape llegó a enviar la especificación del lenguaje a ECMA (European Computer Manufacturer Asociation). En la estandarización el lenguaje se llamó ECMAScript. La versión del lenguaje de Microsoft es JScript. En los últimos años la mayoría de los navegadores han implementado la versión 3 del ECMAScript que tiene gran interoperabilidad. Más recientemente se ha estandarizado la versión 5 del lenguaje por lo que su implementación por parte de los navegadores está en progreso.

Para su funcionamiento el lenguaje debe tener una plataforma, librerías estándar o funciones API para realizar operaciones como la entrada y salida de datos.

### 4.3.1 dijon.js

Dijon es una plataforma de inversión de control (IOC) y de inyección de dependencias (depency injection). En sus inicios fue un intento de migrar Robotlegs y Swiftsuspenders. Si bien dijon.js todavía no llega a la versión 1.0 sin embargo es lo suficientemente estable en su versión 0.62 para poder ser utilizado. Entre sus características se encuentran su independencia de otras librerías y su facilidad para funcionar encima de otras. El estilo de programación puede ser basado en clases o en patrones. Es lo suficientemente flexible para funcionar como enrutador. Su diseño facilita la implementación del modelo de programación MVC. En vez de inyectar dependencias se puede usar un servicio de ubicación de patrones. [26]

El funcionamiento de dijon.js consiste de un objeto en especial el dijon.System que se puede llamar de manera directa o también se puede instanciar. En dijon.js se pueden registrar: Objetos, funciones, clases, puntos de inyección (definen la entidad que recibe los objetos enviados) y llamadas a funciones cuando suceden ciertos eventos.

Entre las ventajas que presenta dijon.js podemos recalcar que nos permite centralizar toda la configuración. Uno de los mayores problemas de las plataformas IOC/DI es resolver los requerimientos de las distintas librerías que crea problemas a la hora de instanciar clases y pasarles sus requerimientos de clases y librerías. En dijon las entidades se comunican unas a otras mediante eventos. Esto da el efecto de que los objetos no saben de los ciclos de vida de los otros sino que simplemente están disponibles para cuando alguna entidad los necesite. Como consecuencia esto conlleva a una configuración centralizada donde los objetos están completamente desacoplados. La ventaja de la configuración centralizada es que sabes dónde buscar las configuraciones de todos los objetos también te permite ver las relaciones entre los distintos objetos en un solo fichero de configuración.

### 4.3.2 q.js

Hay casos en los cuales una función puede no retornar un valor sin bloquearse haciendo uso de un objeto tipo promesa. Un objeto tipo promesa es un objeto que representa el valor de retorno esperado que la función proveerá. Un objeto tipo promesa también puede utilizarse para evitar latencia en un objeto remoto. Con el uso de objetos de tipo promesa también mitiga el efecto de las "pirámides" programáticas. Adicionalmente permite un manejo implícito de errores. Una característica importante es que cuando hay inversión del control (IOC) el uso de promesas permite deshacer la inversión. [25]

Un módulo Q puede utilizarse usando una etiqueta <script>, como un módulo en Node.js y CommonJS, como un módulo AMD, usando un "bower" o un NuGet. Q es interoperable con otras tecnologías como jQuery, Dojo, When.js, WinJS y otras. En la figura 29 se muestra el logo de q.js.


Figura 29. Logo q.js

### 4.3.3 AngularJS

Basado en las experiencias de la creación de aplicaciones como Gmail, Maps y Calendar. Los conceptos clave de angularJs son el uso de plantillas en el lado del cliente. Muchas aplicaciones web crean su código HTML en el servidor., sin embargo AngularJs se diferencia en que envía los datos y las plantillas para que el navegador web los ensamble [27]. Con el modelo de Angular la función del servidor consiste en brindar los datos para las plantillas. AngularJs hace uso de la estructura modelo-vista-controlador (MVC). En AngularJs la vista es el Modelo de Objetos del Documento (DOM), los controladores son las clases de JavaScript y el modelo de datos se almacenan en las propiedades de los objetos. En plataformas anteriores como Rails, PHP o JSP se creaba la interfaz de usuario haciendo uso de código HTML con los datos. Luego con Jquery se extendió el modelo para poder actualizar algunos segmentos del DOM. Sin embargo para actualizar los datos de entrada hay que realizar un esfuerzo considerable AngularJs resuelve esta dificultad a través del enlazamiento de datos (data binding) usando la notación {{ }}. Otro beneficio de AngularJS es la inyección de dependencias que permite que las clases simplemente soliciten lo que necesitan. Para finalizar una de las características más importantes de AngularJs es el uso de directivas que extienden la sintaxis HTML para permitirnos definir mejor la plantilla HTML (por ejemplo: ng-app, ng-repeat). En la figura 30 se muestra el logo de AngularJS.


Figura 30. Logo de AngularJS

## 4.4 VNX

Herramienta de emulación desarrollada por el Departamento de Ingeniería Telemática (DIT) de la Universidad Politécnica de Madrid. VNX es una herramienta de software libre que permita la elaboración de redes virtuales. Se pueden definir distintos escenarios con máquinas ejecutando sistemas operativos distintos (Windoxs, FreeBSD, Linux, enrutadores Dynamips, etc). VNX tiene dos componentes importantes: un lenguaje XML para definir el escenario de virtualización y el programa VNX que interpreta el escenario y administra las máquinas virtuales. Entre los principales beneficios de VNX está en el uso de la librería libvirt y Dynamips. La librería libvirt es la librería principal que permite utilizar las capacidades de virtualización de los diversos sistemas operativos. Dynamips es un software que permite la virtualización de enrutadores de la marca Cisco entre otros. El equipo del proyecto de VNX asimismo provee de máquinas virtuales LXC, KVM ya listas para utilizarse con VNX. [22]

Las máquinas virtuales basadas en el kernel (KVM) son una solución de virtualización para Linux. El kernel es el software que traduce las operaciones de los programas a la CPU, memoria y otros dispositivos hardware. Para poder utilizar esta tecnología de virtualización se requiere tener un procesador Intel VT o AMD-V. La tecnología de Intel VT y de AMD-V permite ejecutar varios sistemas operativos en simultáneo de manera segura y eficiente. Para comprobar si se pueden utilizar máquinas virtuales tipo KVM se puede ejecutar el siguiente comando en Linux:

```
# kvm-ok
INFO: Your CPU supports KVM extensions
INFO: /dev/kvm exists
KVM acceleration can be used
```

Un ejemplo de sintaxis de máquina virtual en VNX:

```
<vm name="client" type="lxc">
        <filesystem
type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-v025
        </filesystem>
        <if id="1" net="Net1" name="eth0">
              <ipv4>192.168.2.62/27</ipv4>
        </if>
        <if id="2" net="Net2" name="eth1">
              <ipv4>192.168.2.65/27</ipv4>
        </if>
        <if id="3" net="virbr0">
              <ipv4>dhcp</ipv4>
        </if>
        <route type="ipv4" gw="192.168.2.33">192.168.2.0/25</route>
        <route type="ipv4" gw="192.168.2.34">192.168.2.128/25</route>
        <forwarding type="ip" />
</vm>
```

Se aprecia las posibilidades de definir interfaces de red, las redes a las que están conectadas, las tablas de enrutamiento necesarias y la imagen de máquina virtual que se va a utilizar. En color rojo se resalta el fichero de máquina virtual a utilizar. En azul se muestran las IP de las distintas interfaces de red. En verde se aprecian las rutas que se utilizaran para acceder a las redes que no están en el mismo segmento de red. En la figura 31 se aprecia el logo de VNX.


Figura 31. Logo de VNX

## 4.5 Resumen

En este capítulo hemos visto las herramientas que se usará para el desarrollo del trabajo. Las herramientas son de tipo software libre lo que permite utilizarlas sin solicitar licencias ni permisos. Las herramientas utilizadas consisten en el reproductor dash.js que nos servirá de plataforma sobre la cual realizar modificaciones. Para efectuar las modificaciones utilizaremos el lenguaje de programación JavaScript que es el lenguaje que se usa en dash.js. El lenguaje JavaScript es uno de los lenguajes más populares en la Internet junto con CSS y HTML. Debido a su popularidad, JavaScript está disponible en muchos dispositivos desde laptops hasta dispositivos móviles como smartphones y tablets. Hemos visto las principales características de dash.js que son su dinamismo a la hora de permitir múltiples tipos de encapsulamiento, reglas para la selección de la tasa de bits y tipos de encapsulamiento. Se menciona también que dash.js usa el estándar H264 para la codificación del vídeo. Para el entorno de emulación del escenario utilizaremos la herramienta VNX que nos permitirá usar máquinas virtuales que tendrán el vídeo y el cliente web. VNX es una herramienta dinámica que utiliza una sintaxis XML para la definición de las distintas máquinas virtuales sus interfaces e interconexiones a través de redes virtuales.

# 5 Presentación del Modelo y Análisis de los Resultados

En el presente capítulo se desarrollará un modelo de emulación de la red con la finalidad de demostrar mejoras en la calidad de servicio a los usuarios haciendo uso de una selección manual de la fuente de vídeo. Para conseguir dicho objetivo se utilizará las herramientas software como dash.js y VNX para implementar el escenario. Asimismo, se extenderá el reproductor web dash.js para que pueda utilizar varias fuentes de vídeo (para un mismo vídeo). Este será el primer paso para poder luego establecer criterios de selección de las CDNs. Para poder probar este concepto de varias fuentes se desarrollará un escenario de virtualización en VNX donde se cargarán máquinas virtuales con los vídeos y el cliente modificado para comprobar que la extensión funcionaba. Adicionalmente se utilizará un shaper que modificará las condiciones de red entre el cliente y el servidor. El shaper tendrá el objetivo de fomentar que las condiciones de red sean malas para provocar el cambio de servidor.

## 5.1 Modelo de Emulación

El modelo de emulación consistirá en un escenario de red donde se tendrá dos servidores que representaran a dos redes distintas de distribución de contenidos. Un cliente con posibilidad de conexión a ambos servidores solicitará un vídeo a uno de ellos y luego procederá a realizar un cambio de servidor de manera manual. Para poder realizar el cambio manual se tendrá que modificar el fichero de manifiesto MPD de los servidores que brindan el contenido DASH. La modificación en el fichero MPD se efectuará mediante la utilización del campo especificado como BaseURL para ingresar las direcciones web de los dos servidores. Una vez que el cliente obtenga el fichero MPD tendrá la información de los segmentos y de la ubicación de los dos servidores. Con las modificaciones a la interfaz y a la parte de control tendremos la capacidad de cambiar de servidor mientras se reproduce un vídeo sin que se detenga o tenga que recargar el buffer. Estas modificaciones son el primer paso para luego realizar un cambio automático en base a las condiciones de red y así brindar un mejor servicio.

### 5.1.1 Escenario de red

A continuación en la figura 32 se muestra el esquema del escenario de red propuesto para el desarrollo del trabajo. El escenario consistirá en un cliente conectado en red a

dos servidores que tendrán el vídeo que el cliente solicitará. Cada servidor representara a una red de distribución de contenidos distinta.

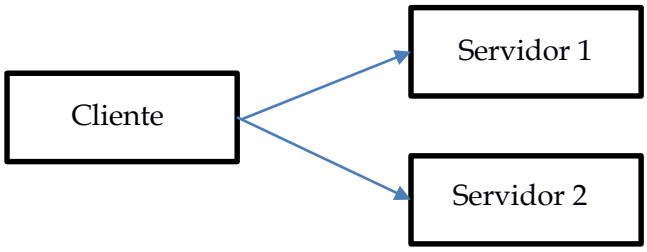


Figura 32. Esquema de escenario de red

Para implementar el escenario de red propuesto se utilizó la herramienta VNX. El escenario consistió de 5 máquinas virtuales con Linux Ubuntu 13.10. Las máquinas tuvieron las funciones de: (a) 1 Host que será la máquina desde la cual se ejecuta el escenario VNX, (b) 1 Cliente que tendrá la aplicación dash.js modificada, (c) 2 Servidores que tendrán los vídeos listos para que el cliente dash.js solicite los segmentos en base al MPD, (d) 2 Shapers que son elementos de red que pueden modificar el ancho de banda de la conexión. A continuación presentamos una figura 33 con el detalle mencionado de los clientes y los servidores.

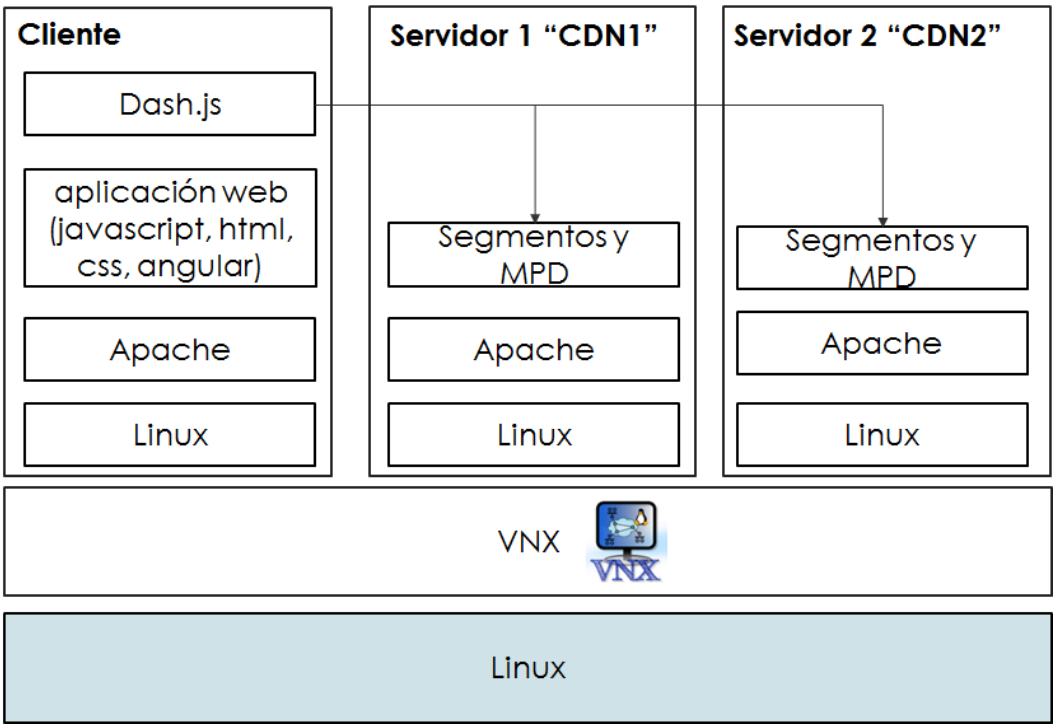


Figura 33. Esquema de los clientes y servidores

En VNX se definieron 4 redes para permitir la interconexión de todos los elementos de red. El fichero de configuración de un escenario VNX consiste en un fichero con etiquetas XML que describen los elementos de red. Los elementos de red tienen propiedades como la red a la que pertenecen con su configuración IP correspondiente, el sistema operativo que ejecutan y la ejecución de programas dentro del elemento de red como un servidor web. El fichero de configuración completa utilizada en el escenario VNX se encuentra en el Anexo 1. La figura 34 muestra en forma gráfica la implementación del escenario de red.

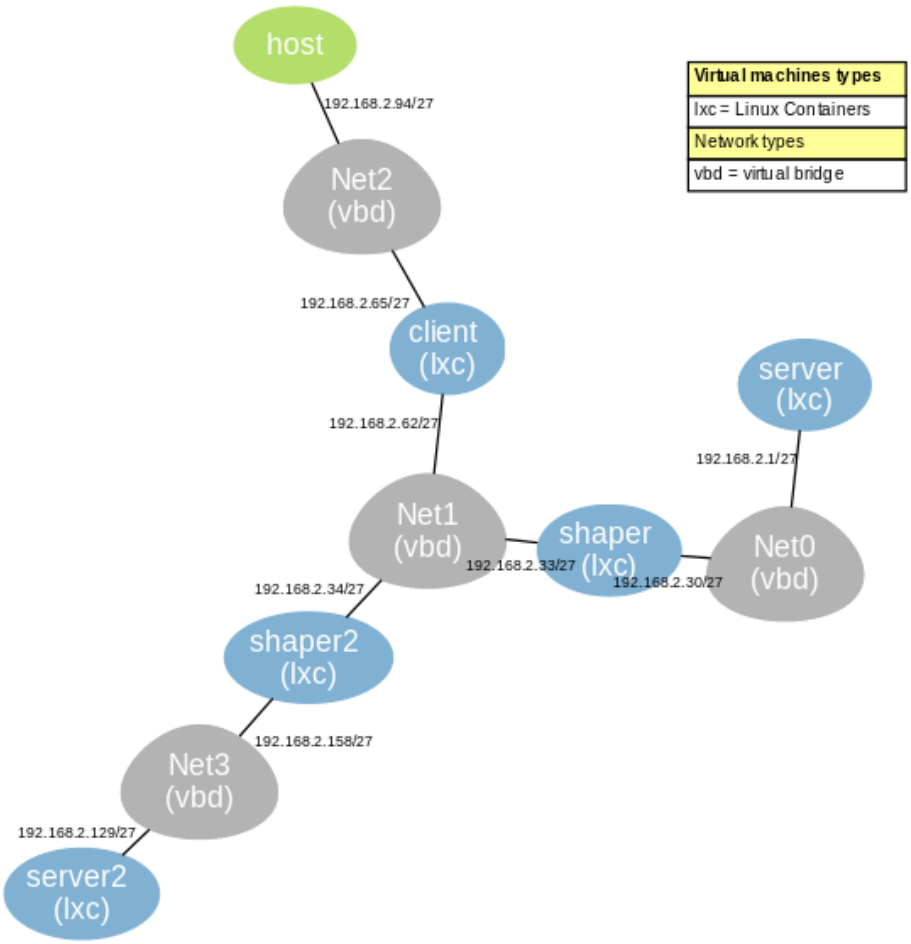


Figura 34. Implementación del escenario de red

La tabla 9 nos muestra el detalle del direccionamiento IP del escenario de red. La máscara de red para todas las redes es: /27 = 1110 0000 = 224.

| Nombre de Red | IP base | Último Byte | Subnet | Dirección Host |
|---|---|---|---|---|
| Net2 | 192.168.2 | 65 | 010 | 00001 |
| | 192.168.2 | 94 | 010 | 11110 |
| Net1 | 192.168.2 | 33 | 001 | 00001 |
| | 192.168.2 | 34 | 001 | 00010 |
| | 192.168.2 | 62 | 001 | 11110 |
| Net0 | 192.168.2 | 1 | 000 | 00001 |
| | 192.168.2 | 30 | 000 | 11110 |
| Net3 | 192.168.2 | 129 | 100 | 00001 |
| | 192.168.2 | 158 | 100 | 11110 |

Tabla 9: Direcciones IP del escenario de red

### 5.1.2 Descripción del escenario desarrollado

Las máquinas virtuales tendrán un servidor web apache para poder brindar los contenidos de los segmentos DASH. Para la generación de los vídeos se han utilizado las herramientas por línea de comandos en Linux:

x264: Para convertir un vídeo y4m (fuente más completa) a h264

MP4Box: Para convertir los vídeos a varias calidades mp4 del estándar DASH y generar los segmentos.

**Operación Normal de la Aplicación Dash.js**

La operación normal de la aplicación dash.js consiste en solicitar un fichero MPD en una ubicación de red específica donde se indica la ubicación de los segmentos de vídeo disponibles. Luego la aplicación dash.js solicita al servidor los segmentos para visualizar el vídeo.

La figura 35 muestra la operación general de la aplicación dash.js.

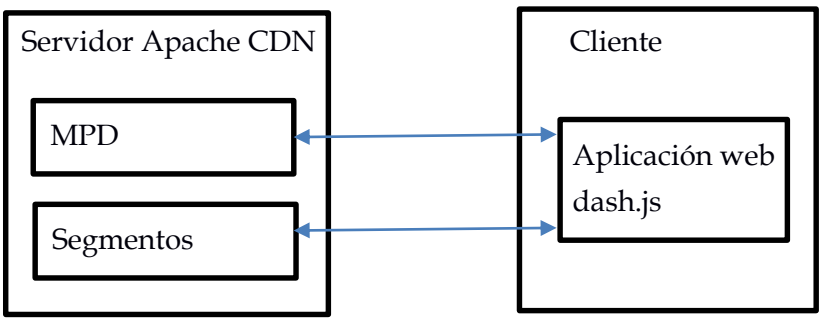


Figura 35. Operación dash.js

**Operación modificada de la aplicación dash.js:**

La nueva capacidad que se le ha agregado a la aplicación web dash.js es la de poder seleccionar los segmentos de un vídeo en servidores distintos de manera manual. Esta modificación hace uso del MPD cargado de manera normal.

La figura 36 muestra la operación modificada del dash.js.

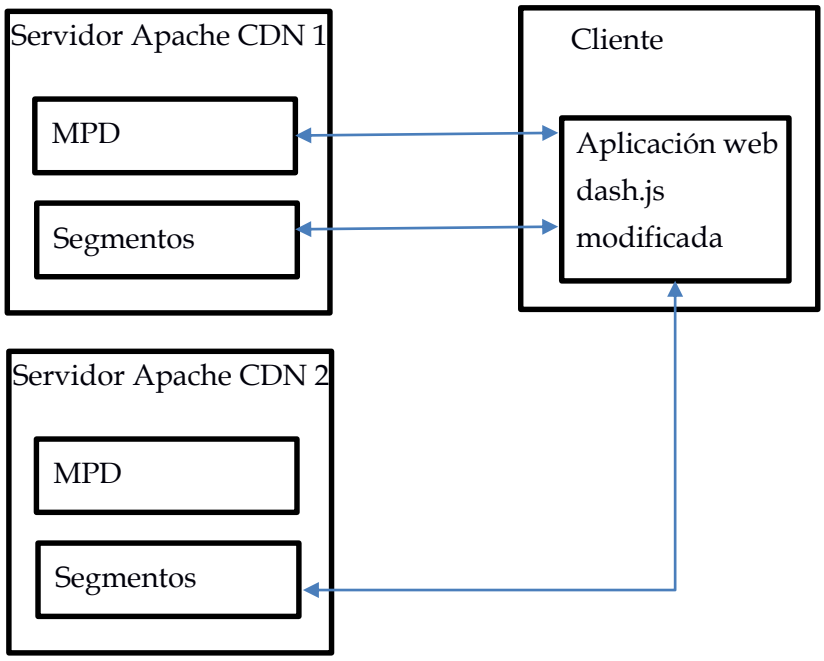


Figura 36. Operación modificada dash.js

Los detalles de los cambios realizados se mencionan en las siguientes secciones.

## 5.2 Modificaciones en el MPD

El fichero MPD es el que contiene la información de los segmentos de vídeo que el cliente puede solicitar. Dentro de la especificación DASH se menciona que se puede incluir varias fuentes para los segmentos de vídeo.

Para esto de definió la etiqueta de BaseURL. Los dos servidores se declaran como se muestra en el código siguiente.

```
  </ProgramInformation>
      <BaseURL>http://server.ltextreme.com/bbbh/</BaseURL>
      <BaseURL>http://server2.ltextreme.com/bbh/</BaseURL>
  …
```

## 5.3 Modificaciones en el controlador

### 5.3.1 Modificaciones en Main.js

Las siguientes modificaciones nos dan la capacidad de seleccionar una CDN de la lista de posibles fuentes que se agregaron en el fichero MPD.

```
$scope.myVar = false;
$scope.showCDN = "Show CDN list";
$scope.selectCDN = function(i){
   console.log(i+" cdn has been selected");
   var a = i;
   var other_manifest = player.getMetricsExt().manifestModel.getValue();
   other_manifest.BaseURL =
player.getMetricsExt().manifestModel.getValue().BaseURL_asArray[i];
   player.getMetricsExt().manifestModel.setValue(other_manifest);
   $scope.currentCdn =
player.getMetricsExt().manifestModel.getValue().BaseURL;
   }

$scope.showCDNList = function(){
   var aux = new String($scope.showCDN);
   if(aux.toString() == "Show CDN list"){
       $scope.showCDN = "Hide CDN list";
       $scope.myVar=!$scope.myVar;};
   if(aux.toString() == "Hide CDN list"){
       $scope.showCDN = "Show CDN list";
       $scope.myVar=!$scope.myVar;};
```

### 5.3.2 Modificaciones en DashHandler.js

La función original para establecer el origen de los segmentos era:

```
var baseURL =
representation.adaptation.period.mpd.manifest.Period_asArray[representation.a
daptation.period.index].
AdaptationSet_asArray[representation.adaptation.index].Representation_asArray
[representation.index].BaseURL, url;
```

Esta función nos dice que el segmento se obtiene de la base URL asociada la información asociada al nivel de segmento del MPD. Esto permite tener diferentes partes de un mismo vídeo replicados servidores distintos. En la siguiente imagen se nos muestra el caso en el que el vídeo A que tiene dos segmentos es distribuido en varios servidores. El MPD en el nivel de segmento nos indicaría que las fuentes posibles para

el segmento 1 serían el Servidor 1 y 2 mientras que para el segmento 2 los servidores disponibles son el 2 y 3. La figura 37 muestras los posibles arreglos de los segmentos de un vídeo dentro de distintos servidores.

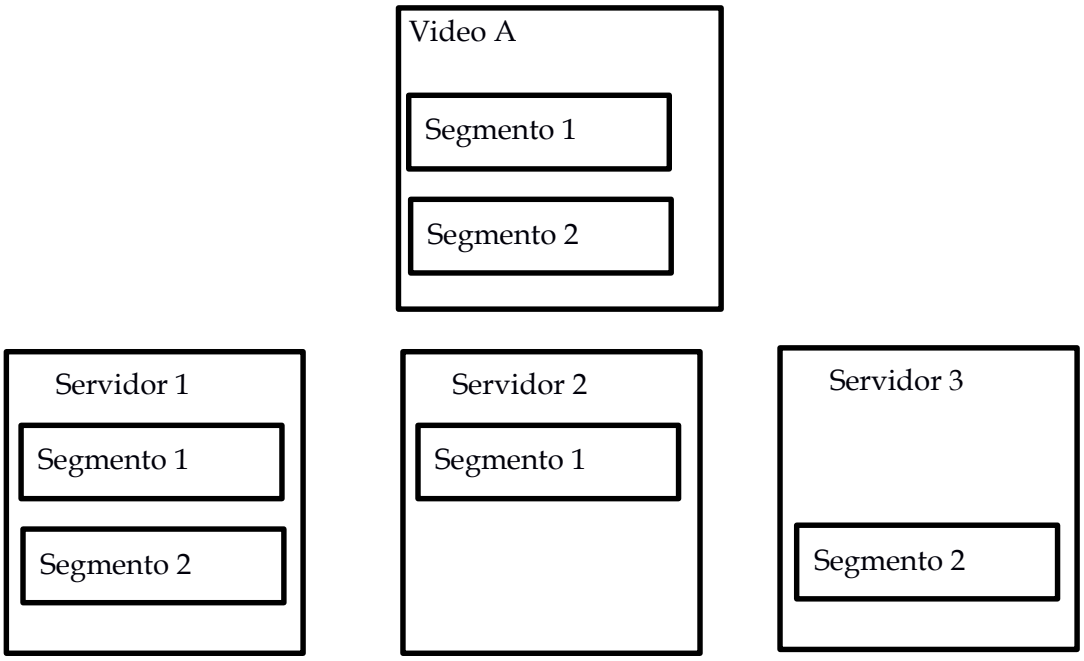


Figura 37. Fuentes a nivel de segmentos

El segmento de código que se muestra seguidamente nos muestra que hemos cambiado la selección del servidor del nivel de segmentos al nivel de global de MPD. En nuestro caso se puede realizar de esta manera porque todo el vídeo está en ambos servidores.

```
getRequestUrl = function (destination, representation) {
var baseURL = representation.adaptation.period.mpd.manifest.BaseURL,url;
            if (destination === baseURL) {
                url = destination;
            } else if (destination.indexOf("http://") !== -1) {
                url = destination;
            } else {
                url = baseURL + destination;
            }
            return url;
        },
```

## 5.4 Modificaciones a la interfaz gráfica

Las modificaciones en la interfaz gráfica tienen por objetivo permitir cambiar de servidor de vídeo. Para esto en primer lugar se debe poder mostrar las opciones de servidores disponibles y el nombre del servidor por defecto. Como paso siguiente se debe se proveer de un mecanismo (botón) para seleccionar la fuente que se quiere. En tercer lugar se debe mostrar la nueva fuente seleccionada. A continuación se presenta el

código HTML que hace posible las funcionalidades mencionadas, también se aprecia el uso de la tecnología de AngularJs con las etiquetas con el prefijo ng y el uso de {{}}.

```
<div class="panel">
   <div class="panel-heading panel-top">
       <div class="btn-group">
          <button type="button" class="btn btn-default" ng-
          click="showCDNList()">
          <span>{{showCDN}}</span>
          </button></br>
       </div>
       <span class="panel-title">Current CDN: {{currentCdn}}</span>
   </div>
   <div class="panel2" ng-show="myVar">
       <div class="panel-body panel-stats">
             <span class="panel-title">CDN list</span>
             <ul>
                   <li ng-repeat="cdn in cdn_items">
                         {{ cdn }}
                         <button type="button" class="btn btn-default"
                         ng-click="selectCDN($index)">
                         <span>SelectCDN [{{$index+1}}]</span>
                         </button>
                   </li>
             </ul>
       </div>
   </div>
</div>
```

El resultado se aprecia en la figura 38.

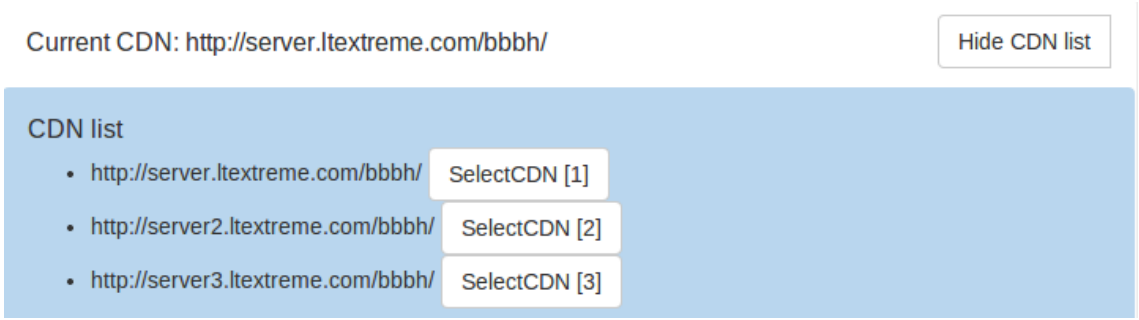


Figura 38. Botones e información adicional incorporada a dash.js

## 5.5 Validez y Confiabilidad de las Modificaciones

Para comprobar el funcionamiento de las modificaciones se utilizó el navegador web Chrome de Google. El navegador Chrome tiene herramientas de desarrollador entre las que se utilizó la pestaña de "Network" que nos muestra las solicitudes HTTP que la página web realiza. En la figura que se muestra a continuación se ve que se realiza el cambio del servidor *server2.ltextreme.com/bbbh* al servidor *server.ltextreme.com/bbbh*

mientras que el vídeo sigue reproduciéndose de manera continua. También se ve que la modificación a la interfaz gráfica muestra correctamente el servidor actual y los posibles servidores que pueden ser seleccionados.

La figura 39 muestra la comprobación de los resultados se ve la interfaz gráfica con el vídeo y la solicitud HTTP GET que realiza el reproductor.

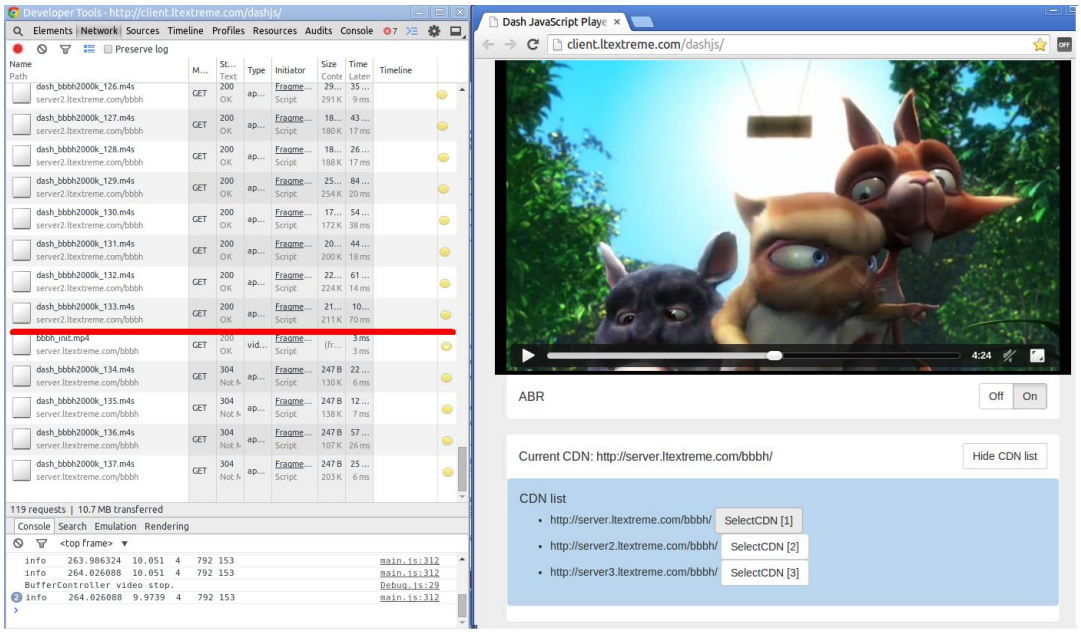


Figura 39. Comprobación de resultados

A continuación se muestra un detalle de la imagen anterior donde se aprecia el cambio de servidor y la continuidad en la solicitud de los vídeos.


Figura 40. Detalle de comprobación de resultados

## 5.6 Escenario de Prueba de las Modificaciones

Para generar un escenario con un ancho de banda variable que simule las características de red que se pueden dar en un entorno de red haremos uso de un "shaper" entre el cliente y el servidor.

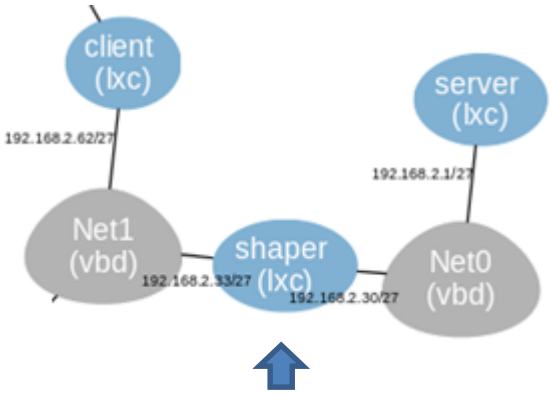


Figura 41. Shaper

El shaper conforma el tráfico TCP haciendo uso de un fichero metrics.txt un ejemplo de dicho fichero se muestra a continuación:

metrics.txt:

```
5,500,2,20,20,0,0
10,1000,1,40,40,0,0
15,500,1,20,20,0,0
20,1000,3,40,40,0,0
25,500,1,20,20,0,0
30,1000,1,40,40,0,0
35,500,3,20,20,0,0
```

La primera columna nos indica el tiempo, la segunda columna nos indica el ancho de banda en Mbps para tráfico UDP, la tercera columna nos indica el ancho de banda en Mbps para el tráfico TCP, las columnas cuarta y quinta son retardos. Finalmente las columnas sexta y séptima representan % de paquetes perdidos.

Como escenario proponemos entre el servidor 1 y el cliente un intervalo de tiempo de 100 segundos con condiciones de red variables para fomentar cambios entre las representaciones seguidos de un ancho de banda reducido que solo permita la representación más baja. Entre el servidor 2 y el cliente se tendrá las mejores prestaciones de red. (En el caso de Hulu que asigna las CDN por probabilidades es un escenario realista, también en el caso de Netflix que no permite cambio de CDN a menos a que se caiga la CDN.)

Obtendremos los resultados de los mensajes de consola que nos da el reproductor dash.js y que tienen la siguiente forma:

```
info      93.933333   0.066667    1     204   55 main.js:312
3
info      93.933333   0.066667    1     204   55 main.js:312
loaded video:Media Segment:96 (200, 669ms, 5270ms) Debug.js:29
info      93.933333   0.066667    1     204   55 main.js:312
Just enough bandwidth available, switch up one. Debug.js:29
info      93.933333   0.066667    2     481   55 main.js:312
Just enough bandwidth available, switch up one. Debug.js:29
info      93.933333   0.066667    2     481   55 main.js:312
Getting the request for time: 94 Debug.js:29
Index for time 94 is 47 Debug.js:29
SegmentList: 94 / 477.17 Debug.js:29
```

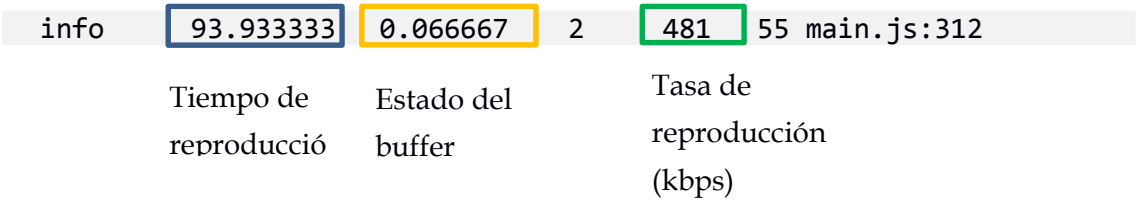


Figura 42. Análisis de los datos de la consola

Juntamos estas trazas de la consola podemos reunir datos para luego graficarlos a lo largo del tiempo y obtener el siguiente gráfico donde se aprecian las condiciones de red variable que hace que el reproductor vaya variando entre las representaciones que tiene disponibles (que son 4: 204 kbps, 481 kbps, 699 kbps y 792 kbps), luego se le proveerá ancho de banda disponible para la representación más baja. Después de dicha situación (en el segundo 160) el usuario decidirá cambiar de CDN y pasará a tener una mejor calidad de manera permanente. A continuación presentamos un gráfico con los datos obtenidos.

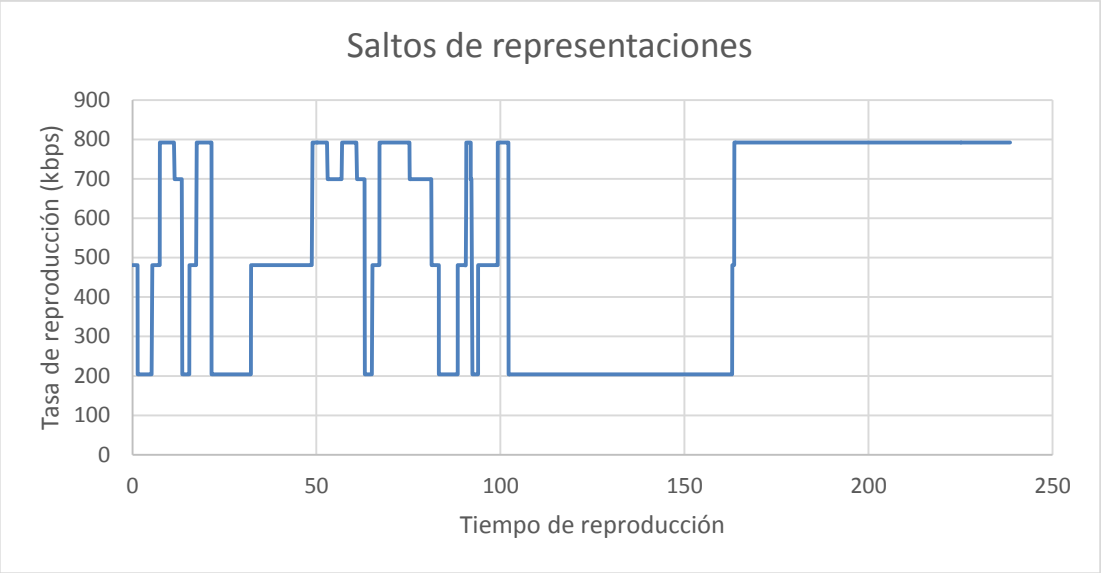


Figura 43. Gráfico de los datos de consola

En el gráfico que mostramos seguidamente mostramos el análisis que podemos hacer del gráfico en el cual se muestran las distintas etapas del escenario y las representaciones que el reproductor ha ido solicitando.

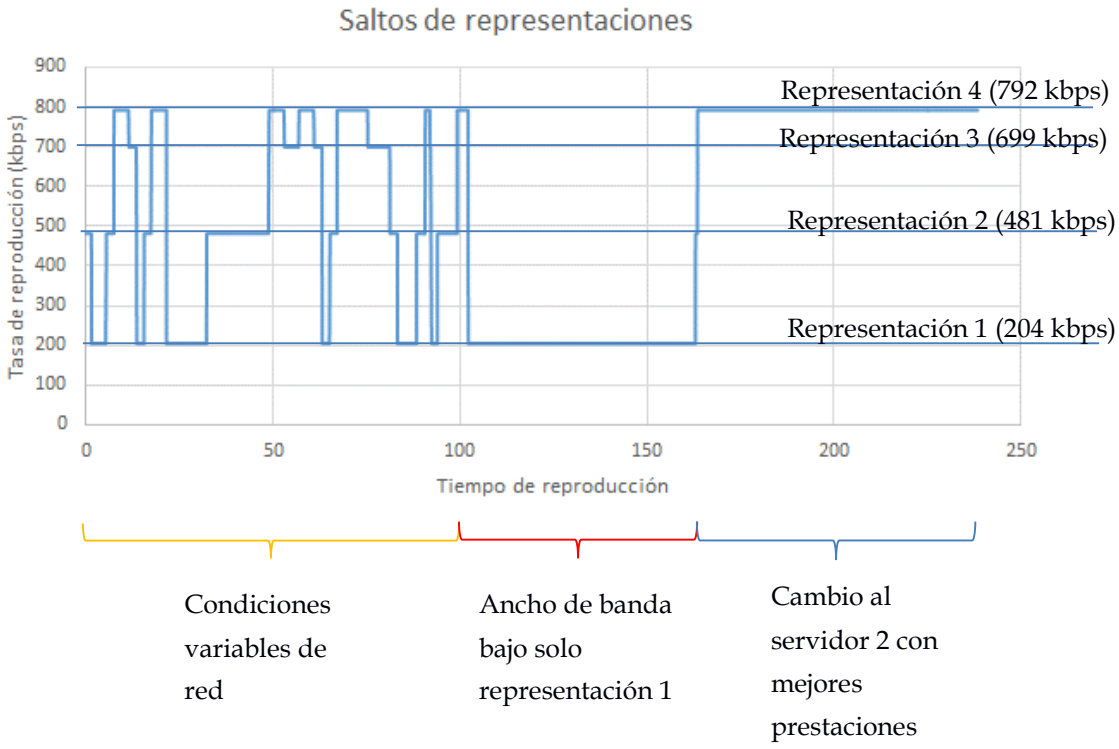


Figura 44. Análisis del gráfico

## 5.7 Resumen

La aplicación dash.js es una aplicación web que hace uso del estándar DASH para la visualización del contenido multimedia. Esta aplicación tiene la inteligencia para pedir los segmentos de vídeo al servidor y poder hacer cambios en las calidades de los segmentos dependiendo de las condiciones de red (debido a que tienen un módulo de monitorización).

Como parte del desarrollo del capítulo se ha modificado la aplicación dash.js para que se pueda seleccionar la fuente del vídeo. Para realizar esta modificación se ha tenido que cambiar la lógica de control de la aplicación, la parte de gráfica de la aplicación al brindar un botón para el cambio y los campos de fuentes de vídeo en el fichero MPD.

Los cambios implementados ha sido puestos a prueba para ello se ha propuesto un escenario de red desarrollado en VNX. El escenario de red consistió en una red con dos fuentes de vídeo y un servidor web donde se cargaron máquinas virtuales con los vídeos y el cliente modificado para comprobar que la extensión funcionaba.

Adicionalmente se utilizó un shaper que degrada las condiciones de red entre el cliente y el primer servidor lo que fomenta al usuario a realizar el cambio al otro servidor de manera manual. El cambio de servidor no reinicio el video sino que continuó de manera transparente.

Como resultados se han obtenido medidas de las trazas de depuración del modo de desarrollo web que nos indican como las calidades van cambiando de acuerdo al ancho de banda y que se podría obtener mejores resultado si se realizará un cambio de fuente. Este caso nos presenta la importancia de tomar medidas de red para luego seleccionar un servidor apropiado.

# 6 Conclusiones

Como conclusiones del presente trabajo se tienen:

La demanda de vídeo en Internet ha ido creciendo y se proyecta que consumirá la mayor cantidad de recursos de red en el futuro próximo. La mejora de las redes cableadas y de la introducción de tecnologías de datos móviles como LTE da la posibilidad a los usuarios de ver vídeos desde sus dispositivos móviles.

En la actualidad los proveedores de vídeo no tienen claro que criterio utilizar a la hora de seleccionar la CDN más adecuada para brindar servicio. Es importante investigar los criterios de selección en base a las condiciones de red para saber que indicadores se podrían utilizar a la hora de seleccionar una CDN.

Es relevante poder cambiarse de CDN cuando dicha CDN no está brindando buenas prestaciones para esto es necesario que los clientes tengan inteligencia y sean parte activa del proceso de reproducción.

En el escenario de red presentado se ve que se puede mejorar la experiencia de usuario al usar servidores de CDN con buenas prestaciones. En el presente trabajo el cambio se ha realizado de manera manual desde el cliente.

Se puede extender el reproductor dash.js para brindar capacidades avanzadas de selección de CDNs. El estándar DASH es relevante como solución de la distribución de vídeos en la Internet debido a la adaptación de la tasa de transferencia. Una funcionalidad importante es la segmentación de los vídeos y la posibilidad de usar distintos servidores de manera transparente. Se ha demostrado la flexibilidad y potencia de DASH para cambiar de servidor sin que el usuario lo perciba manteniendo la continuidad del vídeo haciendo uso del fichero de manifiesto MPD.

Se puede apreciar que la distribución de vídeos en Internet es un problema complejo que abarca la infraestructura de red (acceso), los proveedores de contenidos (empresas como NetFlix y Youtube), las redes de distribución de contenido (como LimeLight y Akamai) y las condiciones de congestión de red que plantean problemas a la hora de brindar los ficheros multimedia a los usuarios.

La extensión de aplicaciones existentes nos permitirá explorar de manera práctica nuevas formas de selección de las fuentes de vídeo (CDNs) usando métricas obtenidas en el cliente.

# 7 Trabajos Futuros

En el presente capítulo indicamos posibles trabajos futuros que continúan en el sentido del presente trabajo de fin de máster.

- Investigar más sobre los criterios de selección de CDNs y lo parámetros a tener en cuenta en dichos criterios
- Implementar una serie de criterios de selección de CDNs en Dash.js
- Implementar otras propuestas de optimización de red como el uso del protocolo ALTO en dash.js
- Investigar el impacto de IPv6 en las tecnologías de distribución de vídeo
- Investigar sobre los criterios y propuestas respecto al uso de redes de distribución de contenidos en redes móviles
- Investigar sobre los criterios de replicación en las CDNs.
- Investigar sobre la posible convergencia del uso de ciertas rutas de red al usar esquemas dinámicos (¿Si sobre una red dada los clientes usan cierto criterio entonces el uso de dicha red alcanzará un estado estable?)
- Investigar sobre las tecnologías y arquitecturas de IPTV que se pueden implementar en la distribución de vídeo de la Internet comercial y viceversa.
- De la misma forma que hay operadores de red virtuales que hacen uso de la infraestructura de red de los operadores de telecomunicaciones se podría dar el caso de proveedores de contenidos virtuales que usen los contenidos de otras empresas, se deberían compartir los servidores caché para los mismos contenidos
- Sobre las páginas de agregación de proveedores de vídeo en las cuales uno busca un vídeo y te dan las opciones, ¿Hay fidelidad por parte de los usuarios?
- Es posible predecir las condiciones de red y sugerir a los usuarios otros proveedores de red a pesar de que estos ofrezcan precios un poco más caros (pero con mayor QoS).
- Uso de sistemas híbridos P2P y Dash para brindar servicios a usuarios cuando la red esta sobrecargada.
- El uso de Scalable Vector Coding (SVC) hace que un vídeo empiece con una calidad base y luego se agregue más información para mejorar la calidad. Si se pierden paquetes la imagen se degrada de manera automática. En este enfoque ¿Es mejor que tener criterios en base a medidas de red y calidades pre-determinadas? (tal vez un enfoque híbrido).
- Las mejores estructuras de CDN todavía son tema de investigación a la vez que la selección de nodos dentro de una CDN.

- La influencia de IPv6 en la distribución de vídeo. Por ejemplo de la dirección IP ya se podría deducir la ubicación de los clientes y servidores. (lo que permite seleccionar el servidor más cercano)

## Bibliografía

[1] The Guardian, *A history of media streaming and the future of connected TV*, [en línea] http://www.theguardian.com/media-network/media-network-blog/2013/mar/01/history-streaming-future-connected-tv [consulta 11/17/2014]

[2] Akamai Technologies, *Akamai Delivers Online Streaming Performance*. [en línea] http://www.akamai.com/html/ms/akamai-delivers-online-streaming-performance.html [consulta 11/17/2014]

[3] David Belson, *Akamai Technologies. The State of the Internet, Q2 2014.* Akamai Technologies, Volume 7 Number 3

[4] Wikipedia High-definition video [en línea] *http://en.wikipedia.org/wiki/High-definition_video*, [consulta 11/17/2014]

[5] Cisco, *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013 - 2018*

[6] *Recommendation ITU-T H.720, Overview of IPTV terminal devices and end systems*, International Telecommunication Union, 2008

[7] *Recommendation ITU-T H.760, Overview of multimedia application frameworks for IPTV services,* International Telecommunication Union, 2009

[8] *Recommendation ITU-T H.770, Mechanisms for service discovery and selection for IPTV services,* International Telecommunication Union, 2009

[9] *Recommendation ITU-T H.264, Mechanisms for service discovery and selection for IPTV services,* International Telecommunication Union, 2009

[10] Begen, A.C.; Akgul, T.; Baugher, M., "Watching Video over the Web: Part 1: Streaming Protocols," *Internet Computing, IEEE* , vol.15, no.2, pp.54,63, March-April 2011

[11] Begen, A.; Akgul, T.; Baugher, M., "Watching Video over the Web: Part 2: Applications, Standardization, and Open Issues," *Internet Computing, IEEE* , vol.15, no.3, pp.59,63, May-June 2011

[12] Cheng Huang, Angela Wang, Jin Li, Keith W. Ross, *Measuring and evaluating large-scale CDNs*, Proceeding of the 8th ACM SIGCOMM conference on Internet measurement, 2008

[13] Spagna, S.; Liebsch, M.; Baldessari, R.; Niccolini, S.; Schmid, S.; Garroppo, R.; Ozawa, K.; Awano, J., "Design principles of an operator-owned highly distributed

content delivery network," *Communications Magazine, IEEE* , vol.51, no.4, pp.132,140, April 2013

[14] Gromov, M.L.; Chebotareva, Y.P., "On optimal CDN node selection," *Micro/Nanotechnologies and Electron Devices (EDM), 2014 15th International Conference of Young Specialists on* , vol., no., pp.136,138, June 30 2014-July 4 2014

[15] S. Wee , J. Apostolopoulos , W. tian Tan and S. Roy "Research and design of a mobile streaming media content delivery network", *Proceedings of IEEE International Conference on Multimedia and Expo*, vol. 1, pp.5 -8 2003

[16] DASH Industry Forum, Survey of European Broadcasters on MPEG-DASH, DASH Industry Forum, 2003

[17] Thomas Stockhammer, Dynamic adaptive streaming over HTTP --: standards and design principles, Proceedings of the second annual ACM conference on Multimedia systems, February 23-25, 2011, San Jose, CA, USA

[18] DASH Industry Forum, Guidelines for implementation: DASH-IF Interoperability Points, 2014

[19] Adhikari, V.K.; Yang Guo; Fang Hao; Varvello, M.; Hilt, V.; Steiner, M.; Zhi-Li Zhang, "Unreeling netflix: Understanding and improving multi-CDN movie delivery," *INFOCOM, 2012 Proceedings IEEE* , vol., no., pp.1620,1628, 25-30 March 2012

[20] Torres, R.; Finamore, A.; Jin Ryong Kim; Mellia, M.; Munafo, M.M.; Sanjay Rao, "Dissecting Video Server Selection Strategies in the YouTube CDN,"*Distributed Computing Systems (ICDCS), 2011 31st International Conference on* , vol., no., pp.248,257, 20-24 June 2011

[21] Adhikari, V.K.; Yang Guo; Fang Hao; Hilt, V.; Zhi-Li Zhang, "A tale of three CDNs: An active measurement study of Hulu and its CDNs," *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on* , vol., no., pp.7,12, 25-30 March 2012

[22] Virtual Networks over linuX, [en línea] http://www.dit.upm.es/vnx

[23] Kernel Based Virtual Machine, [en línea] http://www.linux-kvm.org

[24] Dash Industry Forum, [en línea] https://github.com/Dash-Industry-Forum/dash.js/wiki

[25] Kris Kowal [en línea] https://github.com/kriskowal/q/wiki

[26] Creynders [en línea] https://github.com/creynders/dijon

[27] Brad Green, Shyam Seshadri, AngularJs, 1th ed. O'Reilly, 2013

[28] David Flanagan, JavaScript: The definitive Guide, 6th ed. O'Reilly, 2011

[29] RFC 7230 Hypertext transfer protocol (HTTP/1.1): Message Syntax and Routing

[30] RFC 7231 Hypertext transfer protocol (HTTP/1.1): Semantics and Content

[31] RFC 7234 Hypertext transfer protocol (HTTP/1.1): Caching

[32] Dapeng Wu; Yiwei Thomas Hou; Ya-Qin Zhang, "Transporting real-time video over the Internet: challenges and approaches," *Proceedings of the IEEE* , vol.88, no.12, pp.1855,1877, Dec. 2000

[33] Dapeng Wu; Hou, Y.T.; Zhu, Wenwu; Ya-Qin Zhang; Peha, J.M., "Streaming video over the Internet: approaches and directions," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.11, no.3, pp.282,300, Mar 2001

[33] Dapeng Wu; Hou, Y.T.; Zhu, Wenwu; Ya-Qin Zhang; Peha, J.M., "Streaming video over the Internet: approaches and directions," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.11, no.3, pp.282,300, Mar 2001

[34] Steinbach, E.; Farber, N.; Girod, B., "Adaptive playout for low latency video streaming," *Image Processing, 2001. Proceedings. 2001 International Conference on* , vol.1, no., pp.962,965 vol.1, 2001

[35] RFC 791 Internet Procotol

[36] Cerf, V.; Kahn, R.E., "A Protocol for Packet Network Intercommunication," *Communications, IEEE Transactions on* , vol.22, no.5, pp.637,648, May 1974

[37] Ramanujan, R.S.; Newhouse, J.A.; Kaddoura, M.N.; Ahamad, A.; Chartier, E.R.; Thurber, K.J., "Adaptive streaming of MPEG video over IP networks," *Local Computer Networks, 1997. Proceedings., 22nd Annual Conference on* , vol., no., pp.398,409, 2-5 Nov1997

[38] RFC 2326 Real time Streaming Protocol

[39] RFC 768 User Datagram Protocol

[40] RFC 3550 RTP: A Transport Protocol for Real-Time Applications

[41] P. Deolasse, A. Katkar, A. Panchbudhe, K. Ramamritham, and P. Shenoy. Adaptive push-pull: Disseminating dynamic web data. In Proc. of WWW, pages 265–274, 2001.

[42] Meng Zhang; Qian Zhang; Lifeng Sun; Shiqiang Yang, "Understanding the Power of Pull-Based Streaming Protocol: Can We Do Better?,"*Selected Areas in Communications, IEEE Journal on* , vol.25, no.9, pp.1678,1694, December 2007

[43] MPEG [en línea] http://mpeg.chiariglione.org/

[44] Ding, J.-R.; Yang, J.-F., "Adaptive group-of-pictures and scene change detection methods based on existing H.264 advanced video coding information," *Image Processing, IET* , vol.2, no.2, pp.85,94, April 2008

[45] Alex Zambelli, "IIS Smooth Streaming Technical Overview", Microsoft Corporation 2009

[46] Portable encoding of audio-video objects The Protected Interoperable File Format (PIFF), Microsoft Corporation, 2009.

[47] Apple HTTP live Streaming: http://tools.ietf.org/id/draft-pantos-http-live-streaming-04.txt [en línea]

[48] Schwarz, H.; Marpe, D.; Wiegand, T., "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.17, no.9, pp.1103,1120, Sept. 2007

[49] Jussi Kangasharju, James Roberts, Keith W. Ross, Object replication strategies in content distribution networks, Computer Communications, Volume 25, Issue 4, 1 March 2002, Pages 376-383, ISSN 0140-3664, http://dx.doi.org/10.1016/S0140-3664(01)00409-1.

[50] Jian Ni; Tsang, D.H.K.; Yeung, I.S.H.; Xiaojun Hei, "Hierarchical content routing in large-scale multimedia content delivery network,"*Communications, 2003. ICC '03. IEEE International Conference on* , vol.2, no., pp.854,859 vol.2, 11-15 May 2003

[51] Yun Bai; Bo Jia; Jixiang Zhang; Qiangguo Pu, "An Efficient Load Balancing Technology in CDN," *Fuzzy Systems and Knowledge Discovery, 2009. FSKD '09. Sixth International Conference on* , vol.7, no., pp.510,514, 14-16 Aug. 2009

[52] Dutta, A.; Schulzrinne, Henning, "MarconiNet: overlay mobile content distribution network," *Communications Magazine, IEEE* , vol.42, no.2, pp.64,75, Feb 2004

[53] ISO/IEC 23009-1:2012. Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats

[54] RFC 6265 HTTP State Management Mechanim

[55] RFC 2818 HTTP over TLS

[56] Stockhammer, T.; Luby, M.G., "Dash in mobile networks and services," *Visual Communications and Image Processing (VCIP), 2012 IEEE* , vol., no., pp.1,6, 27-30 Nov. 2012

[57] Netmanias Mircrosoft Silverlight Smooth Streaming workflow http://www.netmanias.com/en/?m=view&id=oneshot&no=5937 [en línea]

[58] Netmanias Apple HTTP Live Streaming http://www.netmanias.com/en/?m=view&id=oneshot&no=5945 [en línea]

# Anexo 1: Fichero XML del Escenario VNX

```
   <?xml version="1.0" encoding="UTF-8"?>
   <!-- ~~~~~~~~~~~~~~~~~~~~ VNX scenarios ~~~~~~~~~~~~~~~~~~~~
   Name: LTExtreme_DASH_MULTICAST_SCENARIO_V1
   Description:    A    scenario    made    of    5    Ubuntu    virtual    machines    (3    hosts:
server.ltextreme.com,server2.ltextreme.com,client.ltextreme.com;
                and 2 routers: shaper.ltextreme.com,shaper2.ltextreme.com) connected through four
virtual networks.
                The host participates in the scenario having a network interface in Net2
                -->

   <vnx xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="/usr/share/xml/vnx/vnx-2.00.xsd">
        <global>
                <version>2.0</version>
                <scenario_name>LTExtreme_DASH_MULTICAST_SCENARIO_V1</scenario_name>
                <automac />
                <vm_mgmt type="none" />

                <vm_defaults>
                        <console id="0" display="no" />
                        <console id="1" display="yes" />
                </vm_defaults>

                <cmd-seq seq="start-mcastmpd">mcastclient,mcastmpd</cmd-seq>
                <cmd-seq seq="start-mcastbbb">mcastclient,mcastbbb</cmd-seq>
                <cmd-seq seq="start-HOmcast">mcastclient,HOmcast</cmd-seq>
                <cmd-seq seq="start-mcastsintel">mcastclient,mcastsintel</cmd-seq>
                <cmd-seq seq="stop-mcast">stop-mcastc,stop-mcasts</cmd-seq>


                <help>
                        <seq_help  seq='mcastsintel'>Start  Flute  Multicast  Sending  Tool  Sintel
Video for DASHJS Player</seq_help>
                        <seq_help seq='mcastmpd'>Start Flute Multicast Sending Tool Sintel Video
for VLC Player</seq_help>
                        <seq_help seq='mcastbbb'>Start Flute Multicast Sending Tool Big Buck Bunny
Video for DASHJS Player</seq_help>
                        <seq_help seq='HOmcast'>Start Flute Multicast Big Buck Bunny Video for
DASHJS Player with HandOvers</seq_help>
                        <seq_help  seq='mcastclient'>Start  Flute  Multicast  Receiving  Tool  in
Client</seq_help>
                        <seq_help  seq='stop-mcasts'>Stop  Flute  Multicast  Sending  Tool  in
Server</seq_help>
                        <seq_help  seq='stop-mcasts'>Stop  Flute  Multicast  Sending  Tool  in
Server</seq_help>
                        <seq_help  seq='stop-mcastc'>Stop  Flute  Multicast  Receiving  Tool  in
Client</seq_help>
                        <seq_help  seq='start-shaperc'>Start  Traffic  Control  from  command
line</seq_help>
                        <seq_help seq='start-shaperf'>Start Traffic Control from a file</seq_help>
                        <seq_help seq='stop-shaper'>Stop Traffic Control shaper</seq_help>
                        <seq_help  seq='clearscenario'>Erase  all  videos  from  the  client
side</seq_help>
                        <seq_help seq='vlc'>Open VLC Client in the HOST</seq_help>
                        <seq_help seq='chrome'>Open Chrome Explorer in the HOST</seq_help>
                </help>
        </global>


        <net name="virbr0" mode="virtual_bridge" managed="no"/>
        <net name="Net0" mode="virtual_bridge" />
        <net name="Net1" mode="virtual_bridge" />
        <net name="Net2" mode="virtual_bridge" />
        <net name="Net3" mode="virtual_bridge" />

        <vm name="server" type="lxc">
                <filesystem   type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-
v025
```

```
                </filesystem>
                <if id="1" net="Net0" name="eth0">
                        <ipv4>192.168.2.1/27</ipv4>
                </if>
                <if id="2" net="virbr0">
                        <ipv4>dhcp</ipv4>
                </if>
                <route type="ipv4" gw="192.168.2.30">192.168.2.0/24</route>

           <forwarding type="ip" />
                <exec seq="on_boot" type="verbatim" ostype="system">
                        service squid3 stop;
                        service apache2 start;
                        rm -r -f /var/www/dashjs;
                        rm -r -f /var/www/dashjs-backup;
                        sudo apt-get install libapache2-mod-perl2;
                </exec>
                <exec seq="mcastmpd" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/mpd/sintel.mpd 1;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:2500 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                 <exec seq="mcastbbb" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/bbb/bbb.mpd 2;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:680 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="HOmcast" type="verbatim">
                /usr/local/bin/hompd2fdt      http://server.ltextreme.com/bbbhm/bbbhm.mpd      2
bbbh/bbbh.mpd;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:2500 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="mcastsintel" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/sintel/sintel.mpd 2;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:650 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="stop-mcasts" type="verbatim">
                pkill flute;
                </exec>
        </vm>

        <vm name="server2" type="lxc">
                <filesystem   type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-
v025
                </filesystem>
                <if id="1" net="Net3" name="eth0">
                        <ipv4>192.168.2.129/27</ipv4>
                </if>
                <route type="ipv4" gw="192.168.2.158">192.168.2.0/24</route>

           <forwarding type="ip" />
                <exec seq="on_boot" type="verbatim" ostype="system">
                        service squid3 stop;
                        service apache2 start;
                        rm -r -f /var/www/dashjs;
                        rm -r -f /var/www/dashjs-backup;
                        sudo apt-get install libapache2-mod-perl2;
                </exec>
                <exec seq="mcastmpd" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/mpd/sintel.mpd 1;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:2500 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                 <exec seq="mcastbbb" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/bbb/bbb.mpd 2;
                cd /var/www;
```

```
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:680 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="HOmcast" type="verbatim">
                /usr/local/bin/hompd2fdt      http://server.ltextreme.com/bbbhm/bbbhm.mpd      2
bbbh/bbbh.mpd;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:2500 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="mcastsintel" type="verbatim">
                /usr/local/bin/mpd2fdt http://server.ltextreme.com/sintel/sintel.mpd 2;
                cd /var/www;
                flute -S -U -T:5 -m:192.168.2.62 -p:4001 -t:2 -r:650 -R:/root/flute.conf -
f:/root/fdt_tsi.xml &amp;
                </exec>
                <exec seq="stop-mcasts" type="verbatim">
                pkill flute;
                </exec>
        </vm>

        <vm name="shaper" type="lxc">
                <filesystem   type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-
v025
                </filesystem>
                <if id="1" net="Net0" name="eth0">
                        <ipv4>192.168.2.30/27</ipv4>
                </if>
                <if id="2" net="Net1" name="eth1">
                        <ipv4>192.168.2.33/27</ipv4>
                </if>
                <if id="3" net="virbr0">
                        <ipv4>dhcp</ipv4>
                </if>
                <route type="ipv4" gw="192.168.2.62">192.168.2.0/24</route>
                <forwarding type="ip" />

                <exec seq="on_boot" type="verbatim" ostype="system">
                        service squid3 stop;
                        service apache2 start;
                        rm -r -f /var/www/dashjs;
                        rm -r -f /var/www/dashjs-backup;
                        rm -r -f /var/www/bbb;
                        rm -r -f /var/www/bbbh;
                        rm -r -f /var/www/bbbhm;
                        rm -r -f /var/www/sintel;
                        rm -r -f /var/www/mpd;
                </exec>
                <exec seq="start-shaperc" type="verbatim">shaperc</exec>
                <exec seq="start-shaperf" type="verbatim">shaperf</exec>
                <exec seq="stop-shaper" type="verbatim">
                tc qdisc del dev eth1 root;
                tc qdisc del dev eth2 root;
                </exec>
        </vm>

        <vm name="shaper2" type="lxc">
                <filesystem   type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-
v025
                </filesystem>
                <if id="1" net="Net3" name="eth0">
                        <ipv4>192.168.2.158/27</ipv4>
                </if>
                <if id="2" net="Net1" name="eth1">
                        <ipv4>192.168.2.34/27</ipv4>
                </if>

                <route type="ipv4" gw="192.168.2.62">192.168.2.0/24</route>
                <forwarding type="ip" />

                <exec seq="on_boot" type="verbatim" ostype="system">
                        service squid3 stop;
                        service apache2 start;
                        rm -r -f /var/www/dashjs;
```

```
                        rm -r -f /var/www/dashjs-backup;
                        rm -r -f /var/www/bbb;
                        rm -r -f /var/www/bbbh;
                        rm -r -f /var/www/bbbhm;
                        rm -r -f /var/www/sintel;
                        rm -r -f /var/www/mpd;
                </exec>
                <exec seq="start-shaperc" type="verbatim">shaperc</exec>
                <exec seq="start-shaperf" type="verbatim">shaperf</exec>
                <exec seq="stop-shaper" type="verbatim">
                tc qdisc del dev eth1 root;
                tc qdisc del dev eth2 root;
                </exec>
        </vm>

        <vm name="client" type="lxc">
                <filesystem    type="cow">/usr/share/vnx/filesystems/vnx_rootfs_lxc_ubuntu-13.10-
v025
                </filesystem>
                <if id="1" net="Net1" name="eth0">
                        <ipv4>192.168.2.62/27</ipv4>
                </if>
                <if id="2" net="Net2" name="eth1">
                        <ipv4>192.168.2.65/27</ipv4>
                </if>
                <if id="3" net="virbr0">
                        <ipv4>dhcp</ipv4>
                </if>
                <route type="ipv4" gw="192.168.2.33">192.168.2.0/25</route>
                <route type="ipv4" gw="192.168.2.34">192.168.2.128/25</route>
                <forwarding type="ip" />
                <exec seq="on_boot" type="verbatim" ostype="system">
                        service squid3 start;
                        service apache2 start;
                        rm -r -f /var/www/sintel;
                        rm -r -f /var/www/mpd;
                        rm -r -f /var/www/dashjs-backup;
                        squid3 -k reconfigure;
                        service apache2 start;
                        chown -R www-data:www-data /var/www/;
                        a2enmod cgid;
                        service apache2 restart;
                        chmod +s /usr/bin/sudo;
                        chmod +x /usr/lib/cgi-bin/hoper.pl;
                        echo "www-data ALL=(root) NOPASSWD:ALL" > /etc/sudoers.d/handover;
                        sudo -f /etc/sudoers.d/handover;
                        sudo apt-get install libapache2-mod-perl2;
                </exec>
                <exec seq="mcastclient" type="verbatim">
                su -l www-data -c 'flute -A -U -p:4001 -t:2 -m:192.168.2.62 -B:/var/www' &amp;
                </exec>
                 <exec seq="stop-mcastc" type="verbatim">
                pkill flute;
                </exec>
                <exec seq="clearscenario" type="verbatim">
                        rm -r -f /var/www/bbb;
                        rm -r -f /var/www/sintel;
                        rm -r -f /var/www/mpd;
                        rm -r -f /var/www/bbbh;
                </exec>
        </vm>

        <host>
                <hostif net="Net2">
                        <ipv4>192.168.2.94/27</ipv4>
                </hostif>
                <route type="ipv4" gw="192.168.2.65">192.168.2.0/24</route>
                <exec  seq="vlc"  type="verbatim">vlc  http://server.ltextreme.com/mpd/sintel.mpd
&amp;</exec>
                <exec     seq="chrome"     type="verbatim">/home/ltextreme/Documents/start-chrome
&amp;</exec>
        </host>
   </vnx>
```

# Anexo 2: Estructura de Ficheros Javascript que Componen Dash.js

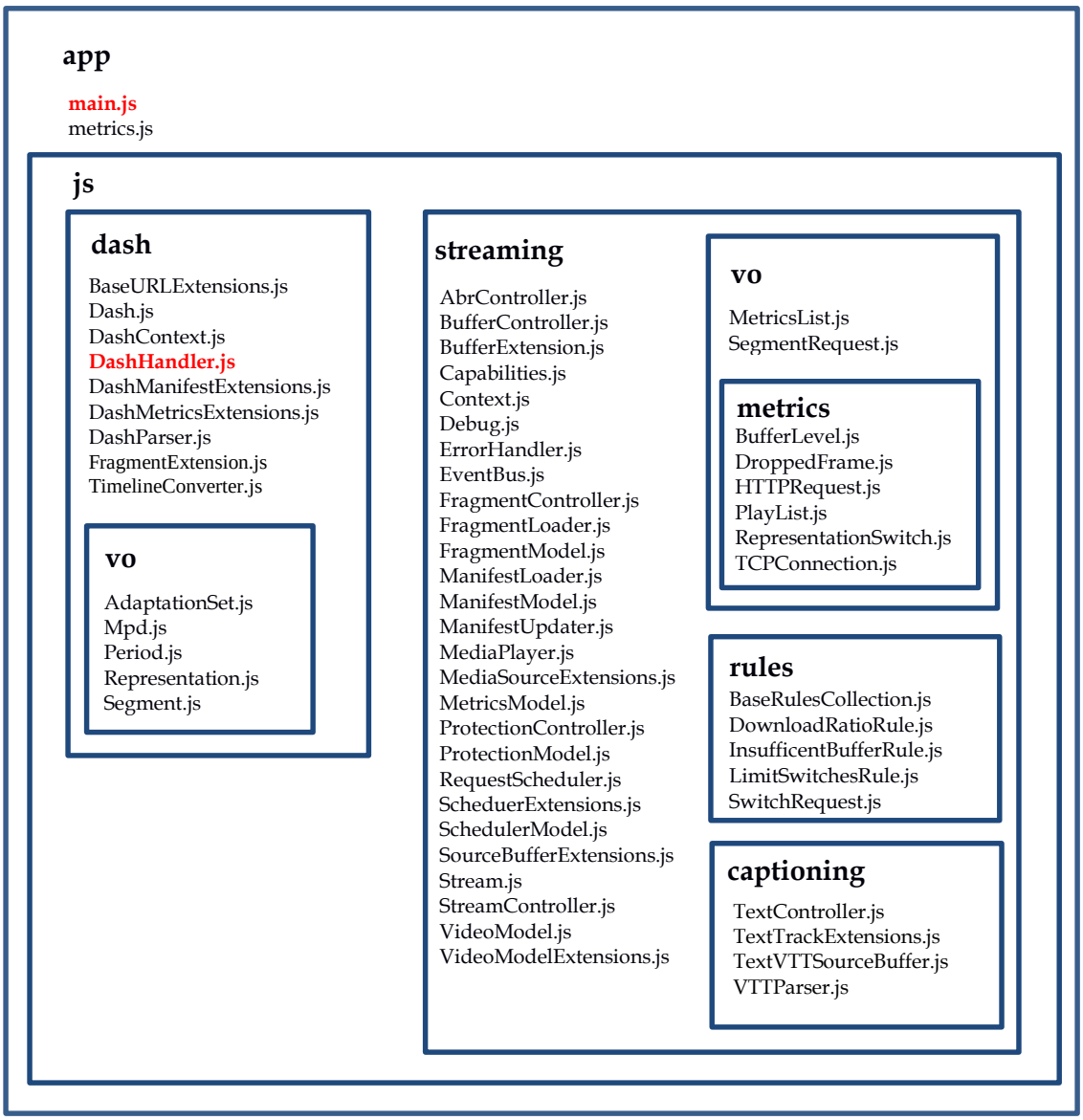


Figura 45. Estructura de ficheros js en dash.js