This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n: 761999

EasyTV: Easing the access of Europeans with disabilities to converging media and content.

Content adaptation using DASH streaming services

EasyTV Project

H2020. ICT-19-2017 Media and content convergence. – IA Innovation action.

Grant Agreement n°: 761999

Start date of project: 1 Oct. 2017

Duration: 30 months

Document. ref.: D4.1

Disclaimer

The EasyTV Consortium consists of the following partners:

	Partner Name	Short name	Country
1	Universidad Politécnica de Madrid	UPM	ES
2	Engineering Ingegneria Informatica S.P.A.	ENG	IT
3	Centre for Research and Technology Hellas/Information Technologies Institute	CERTH	GR
4	Mediavoice SRL	MV	IT
5	Universitat Autònoma Barcelona	UAB	ES
6	Corporació Catalana de Mitjans Audiovisuals SA	CCMA	ES
7	ARX.NET SA	ARX	GR
8	Fundación Confederación Nacional Sordos España para la supresión de barreras de comunicación	FCNSE	ES
9	Unione Italiana dei ciechi e degli ipovedenti	UICI	IT

PROGRAMME NAME:	**H2020. ICT-19-2017 Media and Content Convergence – IA Innovation Action**
PROJECT NUMBER:	761999
PROJECT TITLE:	EASYTV
RESPONSIBLE UNIT:	UPM
INVOLVED UNITS:	CERTH
DOCUMENT NUMBER:	D4.1
DOCUMENT TITLE:	Content adaptation using DASH streaming services
WORK-PACKAGE:	WP 4
DELIVERABLE TYPE:	Prototype
CONTRACTUAL DATE OF DELIVERY:	31-10-2018
LAST UPDATE:	29-10-2018
DISTRIBUTION LEVEL:	PU

Distribution level:

PU = *Public,*

RE = *Restricted to a group of the specified Consortium,*

PP = *Restricted to other program participants (including Commission Services),*

CO = *Confidential, only for members of the LASIE Consortium (including the Commission Services)*

Document History

VERSION	DATE	STATUS	AUTHORS, REVIEWER	DESCRIPTION
v.0.1	20/07/2018	Draft	UPM	Table of Contents definition and document structure
v.0.2	17/10/2018	Draft	UPM	First version of Chapter 1, 2, 3, 4
v.0.3	23/10/2018	Draft	UPM	Second version of Chapter 1, 2, 3, 4
v.0.4	25/10/2018	Draft	UPM	Third version of Chapter 4 First version of Chapter 5 and 6
v.0.5	25/10/2018	Draft	CERTH	Adaptive content that matches user needs/preferences section 4.1
v.0.6	26/10/2018	Draft	ENG	Review
V0.7	29/10/2018	Draft	CCMA	Review
v.0.8	29/10/2018	Final version	UPM	Final version after review

Definitions, Acronyms and Abbreviations

ACRONYMS / ABBREVIATIONS	DESCRIPTION
AAC	Advanced Audio Coding
API	Application Programming Interface
AVC	Advanced Video Coding
DASH	Dynamic Adaptive Streaming over HTTP
DASH IF	DASH Industry Forum
GOP	Group of Pictures
HbbTV	Hybrid Broadcast Broadband TV
HAS	HTTP based Adaptive Streaming
HDS	HTTP Dynamic Streaming
HLS	HTTP Live Streaming
HTML5	HyperText Markup Language v5
HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
MPD	Media Presentation Description
MPEG	Moving Picture Expert Group
MSS	Microsoft Smooth Streaming
PCM	Pulse Code Modulation
URL	Uniform Resource Locator
WebVTT	Web Video Text Tracks
XML	eXtensible Markup Language

Table of Contents

List of Figures

List of Tables

Executive Summary

This document corresponds to the deliverable D4.1 “Content adaptation using DASH streaming services” of the WP4. The deliverable provides not only information about adaptive streaming (basic operation of the technology and the proposed solutions, focusing on MPEG-DASH), but also methods and tools for generating the necessary files for delivering audiovisual contents using MPEG-DASH, and how to access a personalised content on the client application, depending on the needs and preferences of the final users.

The document is divided into six different chapters:

Chapter 1 provides a brief introduction of the project, some of the objectives and the use of MPEG-DASH for the delivery of content. In addition, the different modules on the EasyTV platform used for the content adaptation are also introduced. This chapter provides an overview of the deliverable.

Chapter 2 discusses the different techniques of content delivery in a broadband type scenario, focusing on adaptive streaming, the advantages, the basic operation of the technology and the different proposed solutions.

Chapter 3 provides an introduction to MPEG-DASH, history of this standard and features such as the necessary contents and the DASH client. The tasks done for MPEG-DASH are also mentioned.

Chapter 4 focuses on the use of MPEG-DASH on the EasyTV platform, architecture and workflows. The tests carried out from the generation of the content to the adaptation of the client application and access to the content are also described.

Chapter 5 resumes the next steps and future work, such as the use of other modules defined on the platform to personalise the audiovisual content based on the user profile, as well as the development of other tests related to the use of MPEG-DASH technology and other accessibility services.

Chapter 6 summarizes the conclusions obtained with this deliverable.

1. INTRODUCTION

The EasyTV project is presented with the aim of providing innovative solutions to facilitate universal access to audiovisual services for people with disabilities (mainly visual and hearing disabilities).

In the general architecture of the project, for the distribution of the audiovisual content up to the final users, the MPEG-DASH adaptive streaming technology is used. It is a broadband type scenario where the content is delivered through the Internet network.

Although there are several alternative adaptive streaming solutions, MPEG-DASH is an international standard and it is commonly used in numerous applications such as Netflix or YouTube. The document will discuss the adaptive streaming, the basic operation of this technology and the proposed solutions for audiovisual content delivery. A characteristic of all of these solutions is that they allow to adjust the quality of the audiovisual content, depending on the congestion of the network and the characteristics of the device where the content is played.

For the delivery of the content using MPEG-DASH, the broadcaster has an important role in this scenario. It is necessary to generate for each specific audiovisual content, videos with different quality (typically videos with different resolutions, bit rates and codecs), audios and associated accessibility services. The broadcaster is also the responsible for generating the MPD (Media Presentation Description) manifest file and the initialization dash files associated with the contents. All files are stored on an HTTP (Hypertext Transfer Protocol) web server so that the client application can access to the content through the Internet.

For the personalisation of the content in the client application, the EasyTV platform has the Hyper-Personalisation module that is in charge of adapting the design of the user application and the contents that are played by default to the user (for example, the language of the audio or the accessibility services such as audio description, subtitles or sign language). These services will be those that best satisfy the needs and preferences of the final users. In addition, the users can also manually select on the application the services that they want to watch. This is possible because the MPD includes all the content available for a certain content.

The EasyTV platform will also include a module for modifying the general MPD generated for a specific content. This module will be capable of adding tags with the URLs (Uniform Resource Locators) of files associated with accessibility services. For example, JSON (JavaScript Object Notation) files where the necessary information is found to make the magnification of faces or texts (these are some of the accessibility services contemplated in the project). In this way, when the client application receives the MPD, the application can analyse the whole MPD and can know what accessibility services are available.

In this deliverable, different tests have been done with the MPEG-DASH technology, from the preparation of the audiovisual contents, the generation of the manifest files and the initialization dash files, the modification of the MPDs adding additional accessibility services, and the integration of the DASH client in the application to be able to play the content and select among the different options available.

2. ADAPTIVE STREAMING BASED ON HTTP (HAS)

The delivery of the content through the Internet has been improving during the last years. Initially, it was necessary to download completely the multimedia file to a local hard drive before playing the audiovisual content.

Then the progressive download appeared, a solution where it was not necessary to download completely all the file to start playback the content. An improved alternative of the progressive download is *pseudostreaming*. With this solution it is possible to select any temporal moment and start playing the content when there is enough available information.

In all these cases mentioned above, the content is stored on a local hard drive. With streaming technology, the content is not stored on any hard drive and is played directly. In order to avoid problems related to the interruption of content due to network problems or buffering problems in the devices, the adaptive streaming solutions consist in the creation of fragments of encoded videos stored on HTTP web server. The key concept of this technology is to adjust the quality of the content to the network bandwidth and to the characteristics of the devices.

In adaptive streaming, “intelligence” is on the client side. This means that the client is the most active element within the client-server model (it is known as pull technology) [1]. Therefore, the client is responsible for periodically requesting the bit rate that best satisfied the network conditions and the capabilities of the devices.

This technology uses the HTTP protocol as a mechanism for downloading the audiovisual content. The concept is known as HAS (HTTP based Adaptive Streaming).

In this chapter the basic operation of adaptive streaming is discussed as well as the solutions developed by different standardization organisms.

2.1. Basic operation

In the basic operation of adaptive streaming, there are tasks related to the preparation and generation of the content to make the streaming, and related to the use of the application and the access to the content.

The broadcaster or content provider encodes the video content at multiple bit rates and resolutions (and even with different codecs) and the audio content. Once the files are available, the content is fragmented into small segments (typically from 2 to 10 seconds) so that the client can change quickly between them. Finally, the content is stored on an HTTP web server. There is available a manifest file or a playlist that indicates the client application what files are available.

When starting the streaming of an audiovisual content, the client application receives the manifest file or the playlist (it depends on the adaptive streaming solution used). At the beginning, the client requests the video with the lowest bit rate. In case the download speed is higher than the downloaded bit rate (little congested network), the client will request the video with higher bit rate. If at any time the quality of the network worsens, the client will request the bit rate that is alike to the conditions of the network.

As mentioned above, the client is the most active part of the client-server model, and it is the one that analyses different parameters when requesting the segments. For example, the network conditions related to the bandwidth, the buffer level of the client application, the screen size of the device or the device capabilities (CPU load).

The operation of the adaptive streaming involves significant advantages, both for broadcasters and for end users. It is not necessary to download an important amount of data to start with the playback of the content, it is not necessary to have an important buffer size in the client, and it allows a good quality of experience for both excellent and poor network connections (for poor connections a content with low bit rate will be used and for excellent connections, whenever possible, the highest quality

will be offered). If the technology works correctly, the final user will not experience any problem or interruption in the playback of the content, although network conditions change.

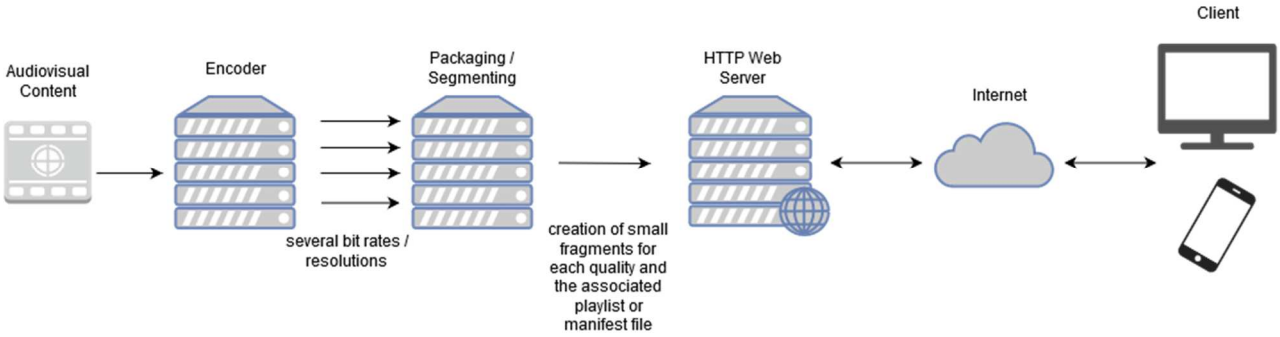

Figure 1: Adaptive HTTP Streaming

2.2. HTTP Adaptive Streaming solutions

Over the last few years, there have been several adaptive streaming solutions proposed by different organisms.

These techniques are improving with the passage of time. All of them are listed. An explanation of each of them is beyond the scope of this deliverable.

- Proprietary solutions:
  - HLS (HTTP Live Streaming) by Apple.
  - MSS (Microsoft Smooth Streaming) by Microsoft.
  - HDS (HTTP Dynamic Streaming) by Adobe.
- International standard solution:
  - DASH (Dynamic Adaptive Streaming over HTTP) by MPEG (Moving Picture Expert Group) and ISO/IEC (ISO, International Organization for Standardization; IEC, International Electrotechnical Commission)

These solutions have the basic operation described in previous section. They are based on HTTP connections between the client and web server for the delivery of audiovisual content, and are based on the idea of using different bit rates and fragments of the content. They work for both live content and content on demand.

The MPEG-DASH standard is described in the following chapter.

3. MPEG-DASH

Among all the solutions mentioned above, MPEG-DASH is the only technology that has been adopted in the HbbTV (Hybrid Broadcast Broadband TV) standard as a technique used for the delivery of broadband content. In addition, MPEG-DASH is also used in other platforms, such as Netflix or YouTube (offering in YouTube 99% of its content with this technology) [2]. The use of this standard is due to the advantages it provides: use of HTTP protocol for content delivery, interoperability with equipment and independence of audio and video coding [3]. Moreover, MPEG-DASH does not define the bit rate control algorithm in the client, the video and audio coding, metadata or a content protection scheme.

The EasyTV project includes the use of the HbbTV 2.0 specification, with the novelty of the multi-device scenarios and the use of second screens (mobile phones, tablets and computers) connected to the main screen. In this type of scenarios, synchronized content can be offered on different devices and the use of MPEG-DASH allows final users to choose accessibility services and personalised content based on their preferences and needs. This is possible because the MPD manifest file includes all the available content associated with a certain service, and the client application can select in each particular case what content it will play.

3.1. Scope

The international MPEG DASH standard was published in April 2012 by ISO/IEC 23009-1. The latest version is in May 2014 [4]. In September 2012 the DASH Industry Forum (DASH IF) [5] was created with the purpose of favouring the use of the standard and to make commercial equipment compatible with the technology. To do this, the forum provides an open source DASH player. It is a dash.js JavaScript library that allows the use of this standard for the playback of the audiovisual content, regardless of the web browser and the audio and video coding [6].

The DASH client has different adaptive algorithms to achieve good performances by selecting the optimum quality of the content according to the conditions of the network and the characteristics of the player. In the applications, the DASH client has been integrated.

3.2. Structure and contents

The basic operation of MPEG-DASH technology has been discussed in section 2.1. It is based on the creation of small fragments of multimedia content stored on an HTTP web server. The content is encoded with different levels of quality and are available, depending on the congestion of the network and the capabilities of the device. On the other hand, this type of streaming needs a smart player (the DASH client) that requests the appropriate fragments to play the content.

MPEG-DASH makes use of the MPD manifest file where the structure of the audiovisual content is described, and provides to the client the location where the specified files are stored. There are two types of MPDs:

- Static MPD: it does not change with the time. Used for video on demand, although it can also be used for live content. It is sent to the client only one time at the beginning of the session.

- Dynamic MPD: when new content is available, the MPD is updated. Therefore, the client must periodically request the MPD because new content could be available. Used for live content.

The structure of the MPD is as follows:

- Periods: the periods describe a part of the content with the start time and the duration of this content. Several periods can be used to divide the multimedia content into different scenes or chapters, and to separate the main content from advertising or other content not related to the main one.

- Adaptation Set: the adaptation set contains a set of streams that belong to the same service. In the simplest case, a period could contain two adaptation sets, one for the video and one for the audio. If there were several audios (for example, with different languages or different audio descriptions), each one would be a different adaptation set. An adaptation set may contain other types of data, such as subtitles or other types of content related to the accessibility. The client uses its preferences to choose the adaptation sets that best satisfied its needs (languages or accessibility services).

- Representations: representations allow an adaptation set to have the same content encoded with different parameters. For example, in the case of the video, different resolutions and different bit rates are usually used. The representations can also be encoded using different codecs, allowing support for clients with different compatible codecs.

- Segments: each representation is formed by one or more segments. Each segment represents each of the parts in which the multimedia content has been divided.

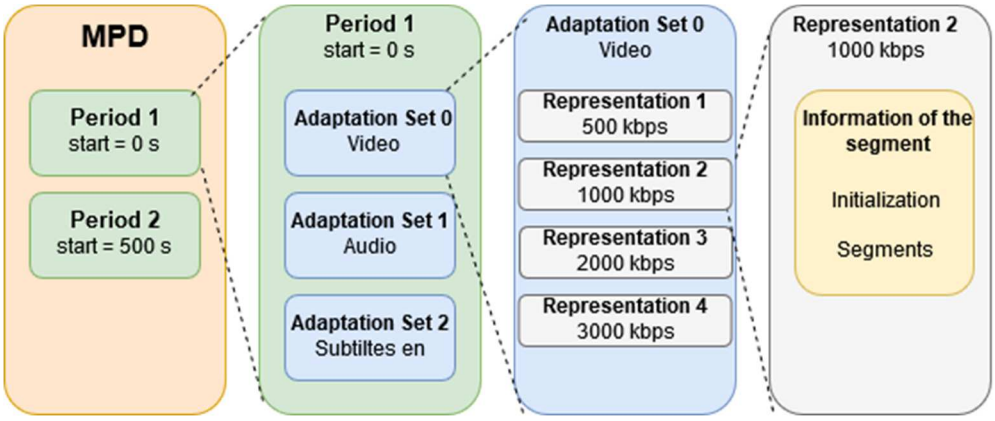

Figure 2: Structure of a general MPD

Due to this structure, MPEG-DASH allows to reuse video component adaptation, providing and efficient tool for reducing the content delivery network costs for content providers or broadcasters on the distribution stage.

3.3. Implementation

To start content delivery using MPEG-DASH, it is necessary to perform the following tasks:

- Preparation of the content. To have the audios available (several audios in case there are different languages or audio descriptions), subtitles, other accessibility services, and videos encoded to different resolutions and bit rates.

- Fragmentation of the content in small segments (typically between 2 and 10 seconds) and the generation of the MPD manifest file.

- Content storage on an HTTP web server.
- Integration of the DASH client in the applications.

To carry out the tests, two tools have been used: FFmpeg and MP4Box. The first is a free software tool that allows to encode and transcode audiovisual content, and it provides also other functionalities. This tool is used by command line, using specific parameters to transcode the contents [7].

The second tool has been used for the generation of the initialization dash files and the MPD manifest file. It is a program available in the open source multimedia project called GPAC [8]. The tool is also used by command line and there are also some parameters that must be used [9].

On the other hand, the dash.js JavaScript library has been used in client applications, which is a DASH client implementation that allows content to be played using this technology and provides access to properties and methods defined in the API (Application Programming Interface).

3.4. MPEG-DASH for HbbTV

The EasyTV project contemplates the use of HbbTV 2.0 specification, published in ETSI TS 102 796 [10], which allows the connection between a hybrid television and second screens, creating a totally different user experience, with synchronized content in all devices. In this way, it is possible to offer different customizable accessibility services in the second screen applications, whereas the main screen application is playing a general content.

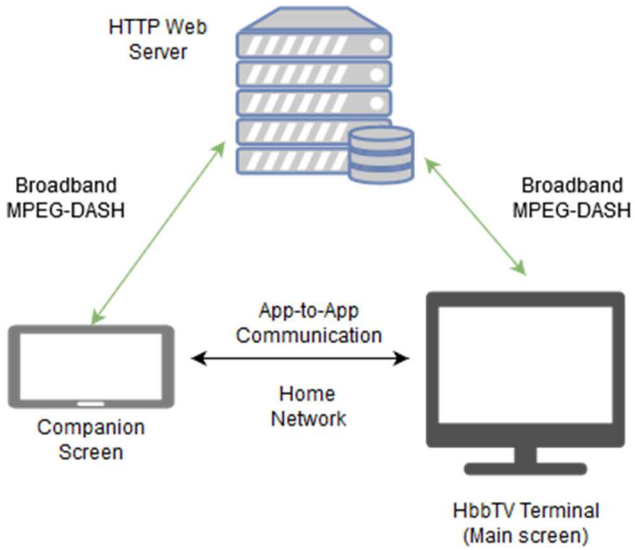

Figure 3: Multi-device scenario in HbbTV 2.0

Both applications request the manifest file stored on a server and, depending on the configuration, select by default the content. In the application, the DASH client is in charge of receiving the MPD, analysing what content is available and selecting the content that is shown in the application. On the other hand, the application itself is responsible for personalising the content (for example, change

the font type of the subtitles, the size, the colour and the background colour).

For the test, two applications were used (one for the main screen and one for the second screen devices) for this project. The application of the main device is a webapp developed in HTML5 (HyperText Markup Language v5) and Angular JS, and the companion screen application is an Android app developed using Apache Cordova, which is a framework that allows to create apps from HTML5, Angular JS and JavaScript. Both applications will communicate with each other with the WebSocket technology using the HbbTV module developed by Fraunhofer [11].

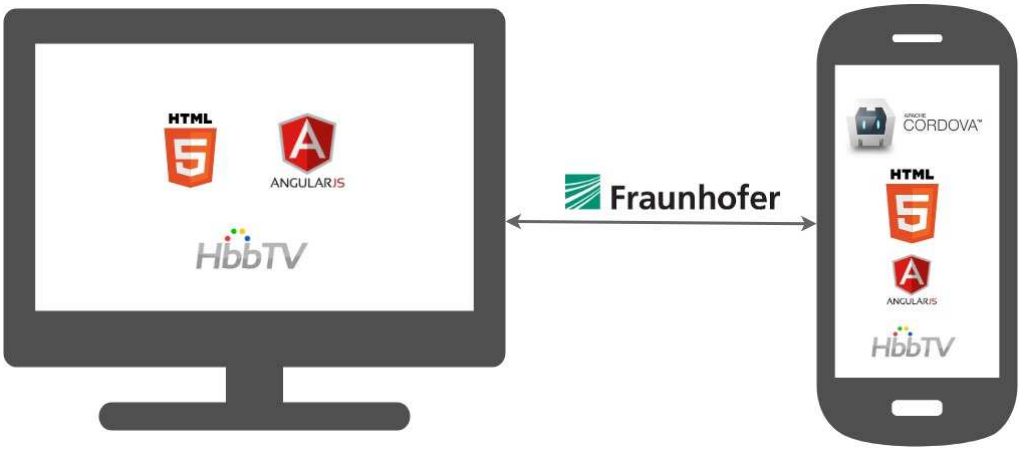

Figure 4: Applications

The HbbTV Fraunhofer module includes some of the features contemplated in the new HbbTV specification, such as the discovery of the devices connected to the same network, the launch of applications between devices and the pairing of applications with a WebSocket connection. Once the connection between the applications is created, it is possible to exchange messages and information between them.

4. USING MPEG-DASH FOR CONTENT ADAPTATION IN EASYTV

4.1. Management of the user needs and preferences

For the management of the user needs and preferences in the client application, the EasyTV architecture includes the Hyper-Personalisation module in charge of adapting the visual design of the user application and the contents that are played by default, depending on the user profile.

The Hyper-Personalization module uses MPD files information for adapting the played content. The information that is considered relevant to the personalization task is the audio and text language, which must best match the user preferred languages. All the other MPDs information are related to the device context and handled by the DASH Client. Under DASH streaming the intelligence lies on the client side. The client selects the appropriate video dimension taking into consideration the screen size and executes adaptive algorithm to adapt the video and audio bandwidth based on network state.

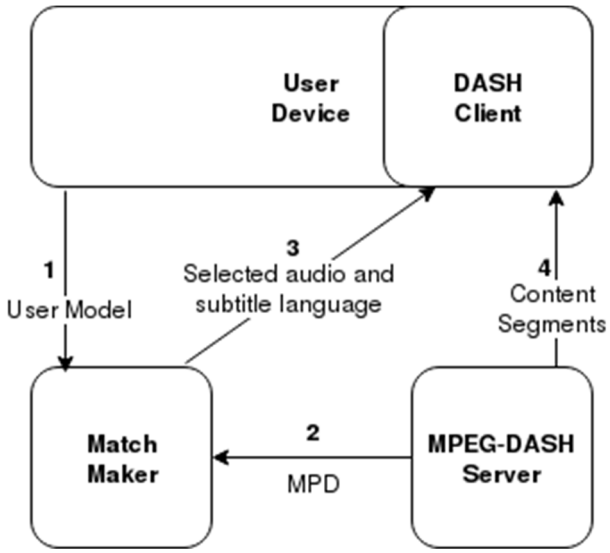

Figure 5. Depiction of the Matchmaker position and interaction

In the personalization process, the user preference for the content audio and subtitle language are extracted from the user model and matched with the available audio and subtitle language listed in the MPD file. For the matching task a set of rules are used in combination with a general inference algorithm. A matched configuration for a user that speaks English would be to have an audio in English OR, in case no English audio available, a proper subtitle. The selected configuration is then communicated to the DASH Client for the adaption to take place.

Table 1. User model relevant attributes to the Matchmaker

Name	Description

{
"preferences" : {
"language_audio" : ""
	The user's preferred audio language, for instance "english"

} }	

{
"preferences" : {
"language_subtitles" : ""
}
}
	The user's preferred subtitles language, for instance "english"

Table 2. MPD file relevant attributes to the Matchmaker

Name	Description
<AdaptationSet contentType="audio" lang=””>	The content audio language.
<AdaptationSet contentType="text" lang=””>	The content subtitle language.

As mentioned, the EasyTV platform updates the content of MPDs files with additional tags and URLs associated with a certain accessibility services, but these are not yet taken into consideration in the personalization process.

In addition to the module discussed above, the application can also store certain information. In the same way, depending on the user profiles, the most appropriate configuration is established for their needs.

For the tests carried out on this deliverable, the Hyper-Personalisation module has not yet been used for the content adaptation. The choice of the content to be displayed in the application as well as the personalisation of it, it is carried out manually by the user.

4.2. Requirements, modular architecture and workflow

In this section it will be described the architecture used in the generation and delivery of MPEG-DASH content, and the workflows that exist.

The broadcaster is responsible, for a certain content, to generate the different qualities of the videos and to have available the audios and accessibility services associated with that content (for example, subtitles or audio descriptions). When the broadcaster has the content available, it generates the fragmentation of the contents and the MPD manifest file.

On the EasyTV platform there is a module that is responsible for modifying the MPD generated by the broadcaster. This module is able to obtain the general MPD and add some extra information related to other additional accessibility services (for example, face magnification and text magnification). Tags are used with the URLs where the information for these services is located. In this way, the application can read the modified MPD and know all the accessibility services that are available for this content. Finally, the modified MPD will be stored on the web server with the rest of the contents.

The client application requests the modified MPD. The application receives and interprets the MPD and knows what content is available for a particular service. The application would check the Hyper-Personalisation module and the information stored in the application, and depending on the user profile, it would establish the configuration of the application. Also the client can select other contents and another type of personalisation.

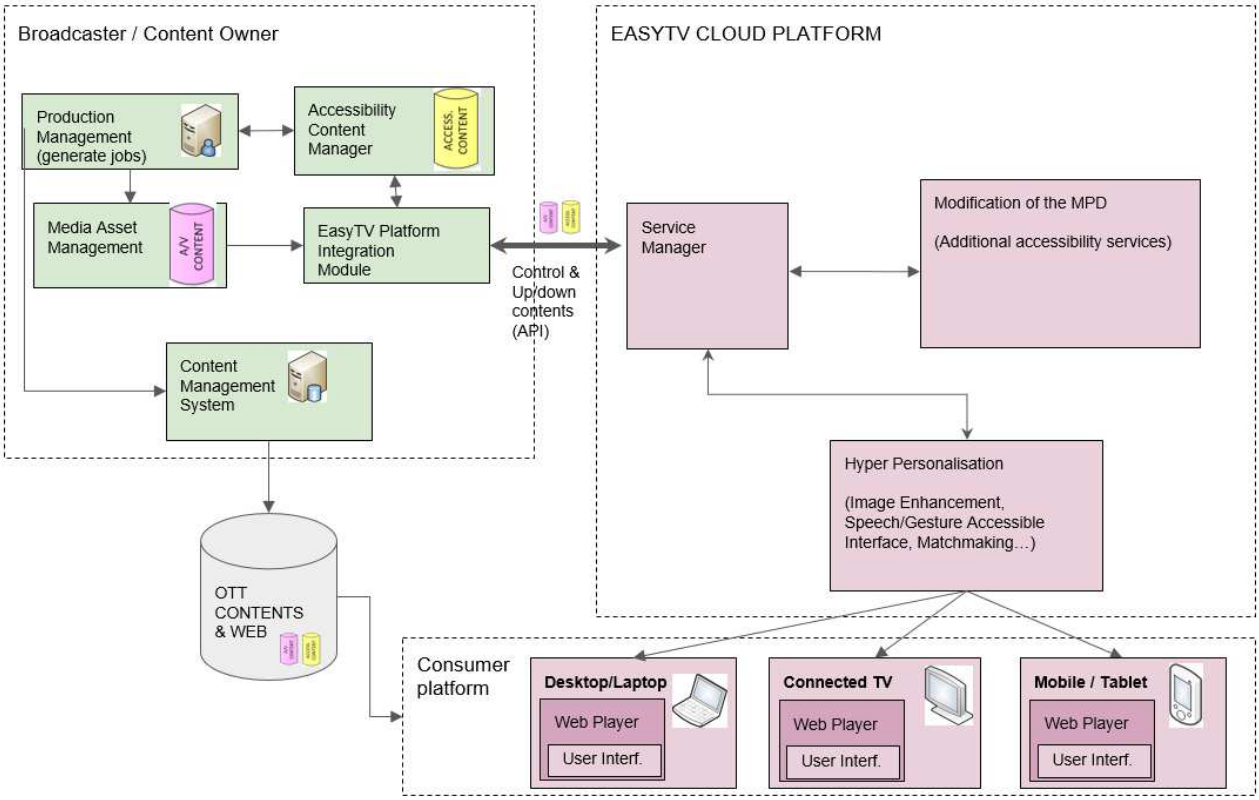

Figure 6: Reduced architecture for content adaptation

4.3. Module development:

In this section it will be explained the tests performed with the MPEG-DASH technology, the different modules discussed above and the use of open source tools used for the preparation of audiovisual content and for the generation of the content for MPEG-DASH.

4.3.1. Phase 1: Preparation of the audiovisual content

Tests have been made with two different contents: a series and a news bulletin. Both contents are from TV3, and provided by CCMA for the development of these tests.

Series: Com si fos ahir (chapter 155)

The initial content of this series is summarized in the following table.

Table 3. Initial content (Com si fos ahir)

Audiovisual Content	Characteristics
	Video Resolution: 1280x720 Frame rate: 25 fps

Content “High Quality” Video + main audio	Codec: AVC Bit rate: 2497 kbps **Audio** Stereo, 44100 Hz Codec: AAC Bit rate: 93 kbps
Content “Low Quality” Video + main audio	**Video** Resolution: 854x480 Frame rate: 25 fps Codec: AVC Bit rate: 994 kbps **Audio** Stereo, 44100 Hz Codec: AAC Bit rate: 93 kbps
Audio + Audio description	Stereo, 44100 Hz Codec: PCM, 16 bits Bit rate: 1536 kbps
Subtitle 1	Lang: cat Codec: WVTT
Subtitle 2	Lang: es Codec: WVTT
Subtitle 3	Lang: en Codec: WVTT

Steps followed:

1. With the FFmpeg tool, for each of the main contents, the audio of the videos has been extracted. In this way, two video files of different quality (with the same characteristics mentioned in the table) and two equal audio files are obtained.

2. With the same tool, the two videos files, the main audio file and the audio description file have been transcoded. In the case of video content, the AVC (Advanced Video Coding) codec was used with the libx264 library of FFmpeg and a GOP (Group of Pictures) size of 50. In the case of audio content, the AAC (Advanced Audio Coding) codec of FFmpeg was used.

3. For the subtitles, the MP4Box tool has been used to convert a WebVTT (Web Video Text Tracks) content into MP4 files and then use it to generate the DASH. Subtitle role and the language are assigned to each subtitle to identify them in the MPD manifest file.

The content obtained is shown in the following table.

Table 4. Content obtained (Com si fos ahir)

Audiovisual Content	Characteristics
Video “High Quality”	Resolution: 1280x720 Frame rate: 25 fps Codec: AVC Bit rate: 1374 kbps
Video “Low Quality”	Resolution: 854x480 Frame rate: 25 fps Codec: AVC Bit rate: 677 kbps
Main audio	Stereo, 44100 Hz Codec: AAC Bit rate: 127 kbps
Audio + audio description	Stereo, 44100 Hz Codec: AAC Bit rate: 131 kbps
Subtitle 1	Lang: ca Codec: WVTT MP4
Subtitle 2	Lang: es Codec: WVTT MP4

News bulletin: Telenoticies TV3

The same steps have been followed for this content. The files resulting from this first stage are shown in the following table.

Table 5. Content obtained (Telenoticies TV3)

Audiovisual Content	Characteristics
Video “High Quality”	Resolution: 1920x1080 Frame rate: 25 fps Codec: AVC Bit rate: 3432 kbps

Video “Low Quality”	Resolution: 960x540 Frame rate: 25 fps Codec: AVC Bit rate: 1087 kbps
Main audio	Stereo, 48000 Hz Codec: AAC Bit rate: 131 kbps

4.3.2. Phase 2: MPD and DASH initial files generation

In the case of the series, the generation of DASH content is carried out with the files shown in Table 3. The MP4Box tool has been used to generate the MPD manifest file and the initialization dash files for each content. In Figure 7 it is shown the command used.

```
gatv@gatv-portege-r930:~$ MP4Box -dash 5000 -rap -frag-rap -profile onDemand -out manifest.mpd
Com_si_fos_ahir_C155_audioX.mp4 Com_si_fos_ahir_C155_audio+audiodescriptionX.mp4:role=alternate
 Com_si_fos_ahir_C155_videoQ1x.mp4 Com_si_fos_ahir_C155_videoQ2x.mp4 subtitle_cat.mp4:role=subt
itle subtitle_es.mp4:role=subtitle subtitle_en.mp4:role=subtitle
```

Figure 7: Command used for generating DASH files

A segment duration of 5 seconds has been chosen (an intermediate value since video on demand it is usually used between 2 and 10 seconds). Segments are forced to start at random access points and the profile is onDemand (for video on demand). In the command it is necessary to indicate the name of the MPD manifest file and list all the contents that will be used.

Once the command has been executed, an initialization dash file is created for each of the contents and the MPD which refers the location of these initialization files. Finally, all generated files are stored in a web server.

```
<?xml version="1.0"?>
<!-- MPD file Generated with GPAC version 0.5.1-DEV-rev5619  on 2018-10-01T15:14:38Z-->
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011" minBufferTime="PT1.500000S" type="static" mediaPresentationDuration="PT0H33M27.02S" profiles="urn:mpeg:dash:profile:
 <ProgramInformation moreInformationURL="http://gpac.sourceforge.net">
  <Title>manifest.mpd generated by GPAC</Title>
 </ProgramInformation>

 <Period duration="PT0H33M27.02S">
  <AdaptationSet segmentAlignment="true" lang="ca" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
   <Representation id="1" mimeType="audio/mp4" codecs="mp4a.40.2" audioSamplingRate="44100" startWithSAP="1" bandwidth="129553">
    <AudioChannelConfiguration schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2"/>
    <BaseURL>Com_si_fos_ahir_C155_audioX_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="850-5717">
     <Initialization range="0-849"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" lang="ca" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="alternate"/>
   <Representation id="2" mimeType="audio/mp4" codecs="mp4a.40.2" audioSamplingRate="48000" startWithSAP="1" bandwidth="131806">
    <AudioChannelConfiguration schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011" value="2"/>
    <BaseURL>Com_si_fos_ahir_C155_audio+audiodescriptionX_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="884-5751">
     <Initialization range="0-883"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" maxWidth="1280" maxHeight="720" maxFrameRate="25" par="16:9" lang="eng" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
   <Representation id="3" mimeType="video/mp4" codecs="avc1.64001f" width="1280" height="720" frameRate="25" sar="1:1" startWithSAP="1" bandwidth="1374311">
    <BaseURL>Com_si_fos_ahir_C155_videoQ1x_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="907-6914">
     <Initialization range="0-906"/>
    </SegmentBase>
   </Representation>
   <Representation id="4" mimeType="video/mp4" codecs="avc1.64001e" width="854" height="480" frameRate="25" sar="1:1" startWithSAP="1" bandwidth="677847">
    <BaseURL>Com_si_fos_ahir_C155_videoQ2x_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="907-6902">
     <Initialization range="0-906"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" group="1" lang="ca" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="subtitle"/>
   <Representation id="5" mimeType="video/mp4" codecs="wvtt" startWithSAP="1" bandwidth="646">
    <BaseURL>subtitle_cat_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="717-5548">
     <Initialization range="0-716"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" group="1" lang="es" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="subtitle"/>
   <Representation id="6" mimeType="video/mp4" codecs="wvtt" startWithSAP="1" bandwidth="650">
    <BaseURL>subtitle_es_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="716-5547">
     <Initialization range="0-715"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" group="1" lang="en" subsegmentStartsWithSAP="1">
   <Role schemeIdUri="urn:mpeg:dash:role:2011" value="subtitle"/>
   <Representation id="7" mimeType="video/mp4" codecs="wvtt" startWithSAP="1" bandwidth="646">
    <BaseURL>subtitle_en_dashinit.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="716-5547">
     <Initialization range="0-715"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
 </Period>
</MPD>
```

Figure 8: Example of a general MPD (Com si fos ahir)

In the MPD, two adaptation sets have been generated for the audio (main audio and main audio with audio description), an adaptation set for the video with two representations (for each of the two qualities) and three adaptation sets for the subtitles (catalan, spanish and english language). It is possible to see in the MPD different parameters such as codecs, bit rates, resolutions or the URLs of the contents.

The procedure followed is the same for the case of the news bulletin content. The files used are those shown in Table 4.

4.3.3. Phase 3: Adding additional information in the MPD

In the EasyTV platform, a module has been implemented to include additional accessibility services in the MPD manifest file. For example, services such as automated face detection and face magnification or automated text detection and text magnification.

The manifest file is an XML (eXtensible Markup Language) document so it is possible to use new different tags and add information within those tags.

A Python module has been implemented. The module obtains the general MPD and adds the tags and URLs associated with certain accessibility services. In the case of the two accessibility services mentioned above, the URLs would be the location of the JSON files that contain the information necessary to provide these services in the client application

```
<accessibility-services>
    <face-detection> ”url_JSON_file_faces” </face-detection>
    <text-detection> ”url_JSON_file_text” </text-detection>
</accessibility-services>
```

Figure 9: Tags for adding additional information in the MPD

The “accessibility-services” tag has been used to contain the additional services. The “face-detection” tag represents the face magnification service and the “text-detection” tag identifies the text magnification service.

The client application receives the modified MPD. Whereas the DASH client is responsible for playing the audiovisual content, the application can read the MPD, the tags, and know what additional accessibility services are available.

```
  </Period>
<accessibility-services>
  <face-detection>http://.....................</face-detection>
</accessibility-services>
</MPD>
```

Figure 10: Example of a modified MPD (Com si fos ahir)

The “accessibility-service” tag is outside the “Period” tag but inside the “MPD” tag. For this case, there is a single additional accessibility service available corresponding to the face magnification. In the figure, the URL of the JSON file is omitted.

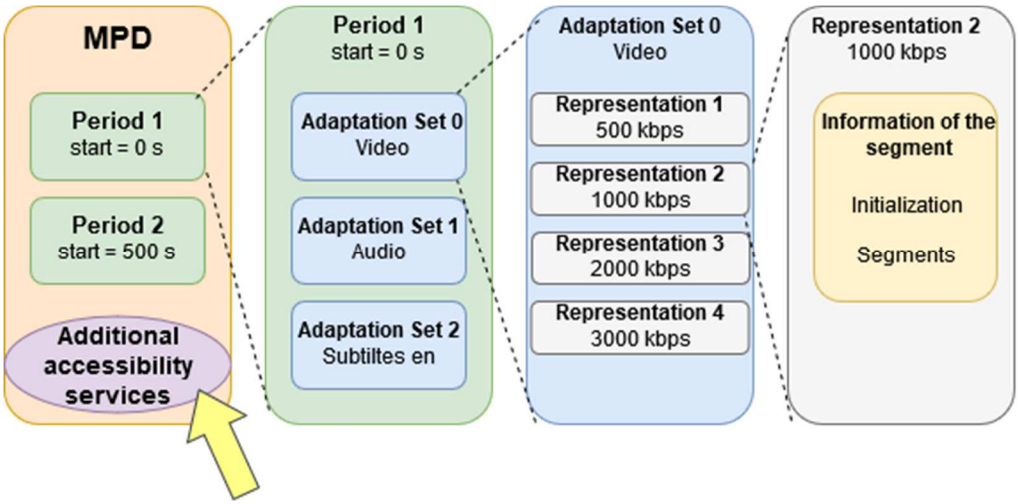

Figure 11: Structure of a modified MPD

4.4. Adaptation of the scenario for MPEG-DASH content presentation

This section explains how EasyTV presents the content, from the MPD file to all the accessibility functions.

The client application uses HTML5 to load a dash.js player, which is able to reproduce MPEG DASH videos. Dash.js automatically manages the download of the MPEG-DASH video fragments, audio and subtitles. As MPEG-DASH uses a bit rate adaptive technique, dash.js will also take care of download different files with different qualities depending on the bandwidth and CPU capacity.

Dash.js provides an API that allows users to play, stop, and change the audio or subtitles tracks, and many other options. These are the API calls used in the application:

- play(): the play method initiates playback of the media. This method will call play on the native Video Element.
- pause(): this method will call pause on the native Video Element.
- setCurrentTrack(idx): this method changes the tracks. Used in the application to change between the language of the subtitles or to change to an audio description track.
- getTracksFor(type): this method returns the list of all available tracks for a given media type (audio, video, text). Used to get all languages available for the audios, audio descriptions and subtitles.
- getCurrentTrackFor(type): this method returns the current track for a given media type (audio, video, text).
- setTextTrack(idx): this method changes the current text track (subtitles). Used in the application to change the language of the subtitles and also to disabled them, using -1 as parameter.
- isSeeking(): the method returns a Boolean that indicates whether the media is in the process of seeking to a new position. Used in the application to pause the video while seeking.
- isReady(): the ready state of the MediaPlayer based on both the Video Element and the MPD

source being defined. Using to play or pause the video while loading the content.

Dash.js manages the standard MPEG-DASH use cases, but the MPD is customized with the accessibility tags commented above. Dash.js is not able to understand these tags, so in order to add these functionalities (additional accessibility services), the application has to read the MPD manifest file as an XML file, and retrieves the tags. The tags contains the URL of the JSON files with the information of these services (for examples, information about faces and texts).

The first step is to retrieve the data from the JSON files. Once the application has the data, it can use the information to magnify the faces or show the recognised text.

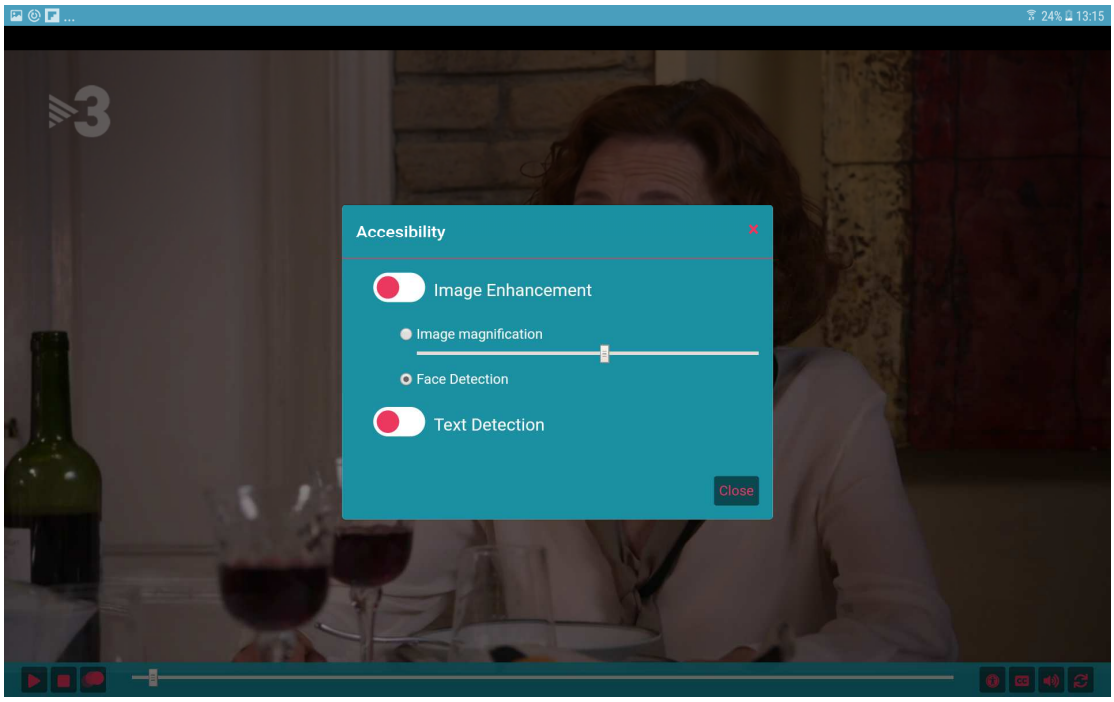

Figure 12: Accessibility pop up

The image enhancement functions match the current video position with the data in the JSON. If the function for face detection is enabled, and there is a face detected for the video position, the application will focus the video on the face with the coordinates provided in the file and will apply a zoom that is also set on the JSON. If there is not a face, the application will disable the zoom. If the functionality for text detection is enabled, and there is a text detected for the video position, the application will pop up a box with the text. The box will have the same style as the subtitles.

Although dash.js takes care of the audio and subtitles, the application implements its own methods to do it. To manage the audios the application uses getTracksFor("audio") to get all the tracks, then differentiate them between main audios and audio descriptions using the "value" attribute and the language attribute defined in the MPD manifest file. The audio descriptions will have the value set to "alternate". Knowing this fact, the application is able to manage the reproduction of the track with the desired language with or without the audio descriptions by calling setCurrentTrack(idx). To retrieve the subtitle tracks, the application uses getTracksFor("text"), then the application allows the user to

change the subtitles by using a custom menu and calling setTextTrack(idx).

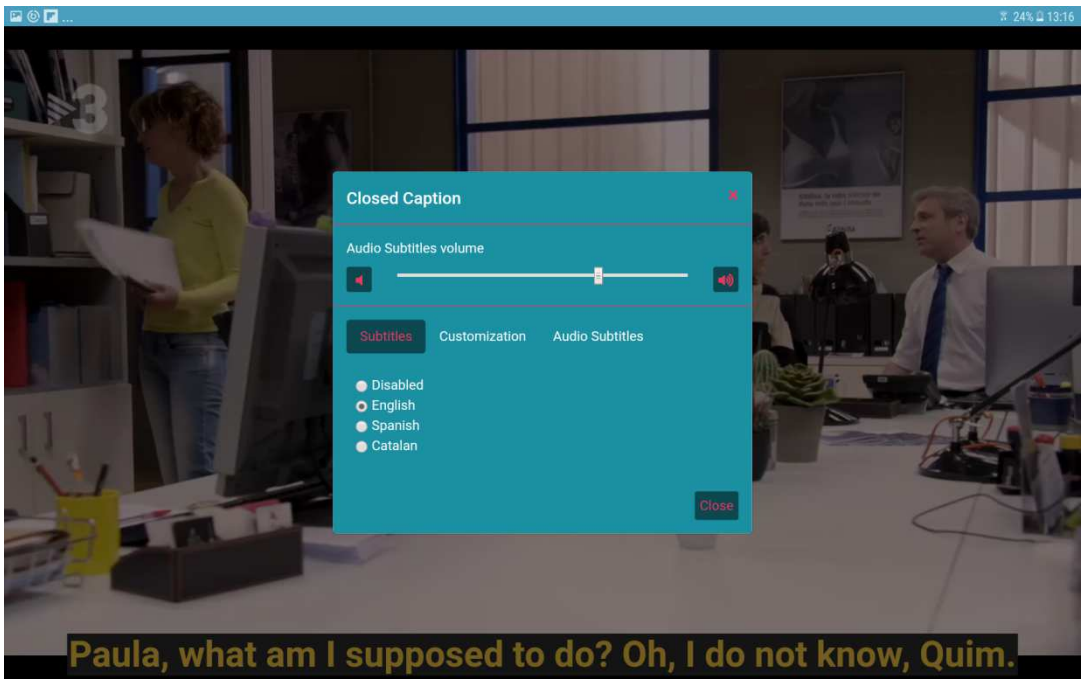

Figure 13: Closed caption pop up

5. NEXT STEPS FOR CONTENT ADAPTATION

In this chapter the next steps and the future work are commented, regarding this task “content adaptation using DASH streaming services”.

In the tests carried out, accessibility services have been included as subtitles and audio descriptions. Also, editing the MPD file, other additional accessibility services related to face detection and text detection. It is intended to include in the MPD new services, sign language and avatars for example.

Now, the personalisation of the accessibility services (in the case of subtitles, the choice of the language, the font colour, the size and the background colour) is done manually by the user. It is intended that the application will receive information from the Hyper-Personalisation module, so that according to the user profile, the visual design of the application and the default contents will be automatically adjusted. The application itself will also store the personalisation information.

6. CONCLUSIONS

In this deliverable, a study of the MPEG-DASH technology has been carried out as an adaptive streaming solution. Using this technology, tests have been done to deliver audiovisual content to a multi-device scenario contemplated in the new HbbTV specification.

The tests start with the preparation of the audiovisual content, the generation of the necessary files to do the streaming with MPEG-DASH, the integration of a DASH client in the applications and the playback of the content and the possibility of changing and personalising the accessibility services. To do this, different open source tools have been used, such as FFmpeg and MP4Box. In addition, a dash.js JavaScript library has been integrated in the applications to play the audiovisual content.

It is important to mention the module implemented in the EasyTV platform to modify a general MPD manifest file and add extra information related to other additional accessibility services. In this way, when the client receives the MPD it knows all services available for that content without having to consult anything else. The EasyTV platform also has the Hyper-Personalisation module that will be used to establish the visual design of the application and the content that is shown by default to the user, based on its needs and preferences.

With this work it is possible to see how the use of MPEG-DASH technology is very useful to provide accessibility services in the second screen devices in a multi-device scenario of HbbTV 2.0.

7. REFERENCES

[1] Dani Marfil, Fernando Boronat, Mario Montagud, Pau Salvador, "End to end platform compatible with HbbTV 2.0 standard for Hybrid TV multi-device". XIII Telematics Engineering Conference, Valencia, September 2017.

[2] Robert Seeliger, Daniel Silhavy, Stefan Pham, Stefan Arbanowski, “Cross-platform ad-insertion using HbbTV and MPEG-DASH”. Asia Pacific Conference on Multimedia and Broadcasting, 2016.

[3] Nemanja Nikić, Marijan Herceg, Vukota Peković, Nenad Šoškić, “System for DASH support verification in HbbTV environment”. IEEE 7th International Conference on Consumer Electronics, Berlin, 2017.

[4] “Information technology; Dynamic adaptive streaming over HTTP (DASH); Part 1: Media presentation description and segment formats”, ISO/IEC 23009-1:2014, May 2014.

[5] DASH Industry Forum, Webpage. [Online]. Available: https://dashif.org/

[6] GitHub: Dash.js, A reference client implementation for the playback of MPEG DASH via JavaScript and compliant browsers. [Online]. Available: https://github.com/Dash-Industry-Forum/dash.js/

[7] FFmpeg. A complete, cross-platform solution to record, convert and stream audio and video. [Online]. Available: https://www.ffmpeg.org/

[8] MP4Box. GPAC, Multimedia Open Source Project. [Online]. Available: https://gpac.wp.imt.fr/mp4box/

[9] DASH Support in MP4Box. GPAC, Multimedia Open Source Project. [Online]. Available:

https://gpac.wp.imt.fr/mp4box/dash/

[10] “Hybrid Broadcast Broadband TV”, ETSI Standard TS 102 796, 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/102700_102799/102796/01.05.01_60/ts_102796v010501p.pdf

[11] GitHub: NodeJS hbbtv module, Fraunhofer FOKUS´s Competence Center Future Applications and Media - FAME. [Online]. Available: https://github.com/fraunhoferfokus/node-hbbtv