Deliverable

Project Acronym:	**IMAC**
Grant Agreement number:	761974
Project Title:	*Immersive Accessibility*

D3.1. Architecture Design

Revision: 2.2

Authors: Chris Hughes (USAL), Peter tho Pesch (IRT), Mario Montagud (i2CAT), Marc Brelot (MSE), Romain Bouqueau (MSE) and Enric Torres (ANG)

Delivery date: M16

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 761974		
Dissemination Level		
P	Public	P
C	Confidential, only for members of the consortium and the Commission Services	

Abstract:
This document describes the architecture design and considerations of the ImAc project. It does this by providing a blueprint describing the architecture for the project, based on the user requirements gathered in D2.3. It explains the objectives and scope of the ImAc system, and details how it fits into the larger ImAc project. The document identifies key requirements, which have a significant bearing on the architecture and how they link directly to the end user requirements. It also provides an architectural representation of the project and establishes the architecture through the use of scenarios, logical and process views and a deployment strategy. It concludes by identifying the key sizing and timing requirements for the platform to be used successfully, as stipulated in the Platform Specification, which will provide the basis for the testing strategy in D3.6.

REVISION HISTORY

Revision	Date	Author	Organisation	Description
1.1	01-01-2019	Chris Hughes	USAL	Template from Iteration 1
1.2	05-02-2019	Chris Hughes	USAL	Updated
2.0	05-03-2019	Chris Hughes	USAL	Implemented all changes as suggested by partners, updated and proof read. Ready for review.
2.1	13-04-2019	Chris Hughes	USAL	Updated following review
2.2	19-07-2019	Zora Schärer	RBB	Updated requirements after pre-pilot 2 results

EXECUTIVE SUMMARY

This document is presented in two iterations and describes the technical architecture of the ImAc project. As the first iteration it is a working document providing a blueprint for the technical implementation of the project, and has been updated after the initial prototype implementation and testing phase to reflect changes needed to improve the platform.

The technical architecture is directly based on the user requirements established in WP2, specifically the platform specification (D2.3) that identifies the essential system requirements needed to ensure a successful implementation. In turn it feeds directly into the Integration and Testing Report (D3.6), which is designed to evaluate the success of each phase of implementation.

Chapter 1 provides an overview of this document, describes the objectives and scope of the technical architecture, and details how it fits into the larger ImAc project.

Chapter 2 describes the Goals and Constraints of the system architecture. These are the key requirements, which have a significant bearing on the architecture and they link directly to the end user requirements defined in the requirements table in D2.2. Satisfying these needs is essential to developing a successful framework for the ImAc project.

Chapter 3 provides an architectural representation of the project and establishes the architecture in terms of usage scenarios, logical and process views and a deployment structure.

Chapter 4 describes the existing resources and libraries, which will be used within the implementation of the project, where they are used within the architecture, and how they will be integrated with.

Chapter 5 discusses the size and performance requirements of the implementation. This details the key sizing and timing requirements for the platform to be used successfully, as stipulated in the Platform Specification, which will provide the basis for the testing strategy.

Chapter 6 concludes the document with a summary of the technical architecture.

CONTRIBUTORS

First Name	Last Name	Company	e-Mail
Chris	Hughes	USAL	c.j.hughes@salford.ac.uk
Mario	Montagud	i2CAT	mario.montagud@i2cat.net
Marc Romain	Brelot Bouqueau	MSE	marc.brelot@gpac-licensing.com
Enric	Torres	ANG	enric@anglatecnic.com
Peter	tho Pesch	IRT	thopesch@irt.de

CONTENTS

LIST OF ACRONYMS

Acronym	Description
AD	Audio Description
ACM	Accesibility Content Manager
AST	Audio Subtitles
AWS	Amazon Web Services
CDN	Content Delivery Network
CM	Content Manager
CMAF	Common Media Application Format
DASH	Dynamic Adaptive Streaming over HTTP
DVB	Digital Video Broadcasting
EBU	European Broadcast Union
FOV	Field of View
FTP	File Transfer Protocol
HbbTV	Hybrid broadcast Broadband TV
HMD	Head Mounted Display
HUR	Home User Requirement
IMSC	Internet Media Subtitles and Captions
ISOBMFF	ISO Base Media File Format
ISP	Internet Service Provider
MPEG	Moving Picture Experts Group
OBA	object based audio
PUR	Professional User Requirement
SL	Sign Language
ST	Subtitles
TT	Timed Text
TTML	Timed Text Markup Language
WebVTT	Web Video Text Tracks Format

1. INTRODUCTION

1.1. Purpose of this document

The final goal of WP3 is to define and implement a platform integrating different components of the production chain, including accessible content management, packaging and distribution, customisation of the experience, and display of immersive and adapted content. The design will be fed by requirements gathered in WP2 (T2.2. and T2.3). This includes:

- To design the architecture of a robust platform capable to integrate all the components developed in the project.
- To design and implement a content management component facilitating access to multiple content formats and its storage.
- To adapt and integrate a real-time process to effectively encode multiple streams from inclusive content (i.e. subtitles, audio description and sign language) into full omnidirectional video.
- To design and implement a delivery chain that can process the input from Production (especially for the Accessibility and Immersive sides) and make them available to the player using standard technologies.
- To design and implement a player (including clients and libraries) required to display omnidirectional video across devices (TV, second screen and HMD) maintaining coherence, synchronization, and enabling interaction and personalisation features.
- Validate development in semi-open pilots and large-scale pilots.
- Disseminate and communicate the WP outcomes among other researchers and industry stakeholders.

This document provides the technical architecture, which describes the blueprint for each of the subsequent stages of WP3.

1.2. Scope of this document

This document will provide a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions, which have been made on the system based on User requirements (D2.2), and Platform Specifications (D2.3) defined in WP2.

The document follows a logical progression consisting of four main sections:

- Chapter 2: Architectural Goals and Constraints
- Chapter 3: Architectural Representation
- Chapter 4: Dependencies
- Chapter 5: Size and Performance

Chapter 2 describes what we need to do. This is based on the user requirements gathered in D2.3. Chapter 3 explains how we are going to do it. This will contain the blueprint for the ImAc system. Chapter 4 details the tools we are going to use to make the platform work. Finally Chapter 5 discusses the size and performance requirements of the implementation of each component of the ImAc system.

1.3. Status of this document

This is the second iteration of this document, which has been revised following an initial development and testing cycle. It has been updated to reflect the lessons learned from the initial pilot study and therefore the takes into account the updated user requirements.

1.4. Relation with other ImAc activities

The PERT diagram illustrates the relation between D3.1 and the other ImAc activities. The Architecture design is built based upon the findings of D2.3 Platform Specification and it feeds into the implementation of both the Accessibility Services and the Immersive Platform.

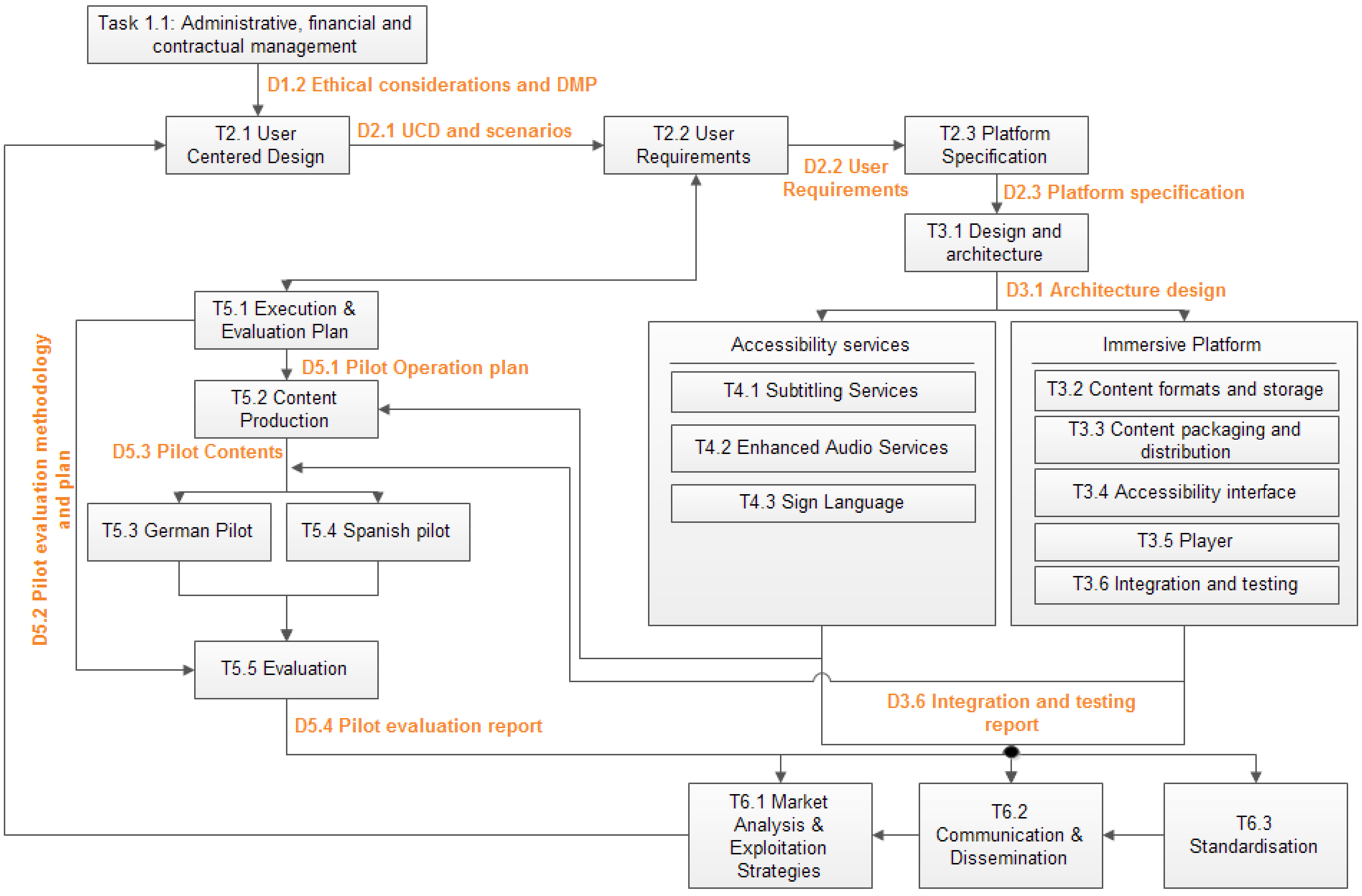

Task 1.1: Administrative, financial and contractual management
D1.2 Ethical considerations and DMP
T2.1 User Centered Design
D2.1 UCD and scenarios
T2.2 User Requirements
D2.2 User Requirements
T2.3 Platform Specification
D2.3 Platform specification
T3.1 Design and architecture
D3.1 Architecture design
T5.1 Execution & Evaluation Plan
D5.1 Pilot Operation plan
T5.2 Content Production
D5.3 Pilot Contents
D5.2 Pilot evaluation methodology and plan
T5.3 German Pilot
T5.4 Spanish pilot
T5.5 Evaluation
D5.4 Pilot evaluation report
Accessibility services
T4.1 Subtitling Services
T4.2 Enhanced Audio Services
T4.3 Sign Language
Immersive Platform
T3.2 Content formats and storage
T3.3 Content packaging and distribution
T3.4 Accessibility interface
T3.5 Player
T3.6 Integration and testing
D3.6 Integration and testing report
T6.1 Market Analysis & Exploitation Strategies
T6.2 Communication & Dissemination
T6.3 Standardisation

Figure 1 - PERT Diagram illustrating the relationship between D3.1 and other ImAc activities.

2. Architectural Goals and Constraints

2.1. Key Requirements

D2.2 followed a user centric design approach to understand the needs of users and provided a detailed description of the user requirements for the ImAc platform. The key requirements and system constraints, which have a significant bearing on the architecture, are listed in Table 1a (Home Users) and Table 1b (Professional Users). These link directly to the end user requirements defined in the requirements table in D2.2 with a priority of 'Must' or 'Should'. Satisfying these needs is essential to developing a successful framework for the ImAc project.

Table 1a - Key requirements for the home user taken from D2.2 that are required for a successful implementation.

Version	Title	Description	Prioritization
HUR.2.4.1	Access to Audio Description	The user can use a visual menu to access the service. It should be large, with a black background and white text for the highest possible contrast. Yellow will be used to highlight the choice of the user.	MUST
HUR.2.6.0	Multiplatform player for desktop, mobile phone (cardboard supported, gyroscope sensor based), TV, head mounted display	The user can start, pause, resume, forward or rewind the omnidirectional media with a graphical user interface.	MUST
HUR.2.8.0	Subtitles always on main Screen	Subtitles are always presented on the main screen, i.e. users do NOT want the subtitles to be delivered on an additional (companion) screen when accessing content	SHOULD
HUR.2.9.0	Playback of audio description	The player enables the audio description to be synchronized with the main audio track	MUST
HUR.2.10.0	Player support for screen reader functionality	The player provides spoken feedback of all interface controls	MUST
HUR.2.10.1	Player support for screen reader functionality	The user can control the volume of the spoken feedback	SHOULD
HUR.2.13.0	User settings persistence	The player retains user preferences between users	SHOULD
HUR.2.16.0	Switch on/off signer	The user has the possibility to switch on/off the signer with a graphical user interface.	MUST
HUR.2.17.0	Selection of personalization options for signer	The user has the possibility to activate and deactivate different personalization options with a graphical user interface.	MUST

HUR.2.18.0	Accessibility interface signer - basic presentation mode	There is one basic presentation mode for the signer, which is always available for the user on any device. This mode presents it as follows: The signer video has a fixed position on the bottom right area of the field of view and the user decides what direction he/she wants to look.	MUST
HUR.2.19.0	Accessibility interface signer - position in viewing field	The user can select between a predefined set of different horizontal positions for the signer in the "basic presentation mode".	MUST
HUR.2.21.0	Accessibility interface signer - speaker location indicator	The signer is always positioned in the user's field of view according to personalization settings and an arrow under the signer indicates the position of the current speaker	SHOULD
HUR.2.21.2	Accessibility interface signer - speaker location indicator	The signer is always positioned in the user's field of view according to personalization settings and radar interactively indicates the position of the speaker by positioning a dot inside the radar field in relation to the viewer's orientation	SHOULD
HUR.2.24.0	Switch on/off subtitles	The user has the possibility to switch on/off subtitles with a graphical user interface.	MUST
HUR.2.25.0	Select subtitle tracks	The user has the possibility to select different subtitle tracks with a graphical user interface.	MUST
HUR.2.26.0	Selection of Personalization options for subtitles	The user has the possibility to activate and deactivate different personalization options with a graphical user interface.	MUST
HUR.2.27.0	Accessibility interface for subtitles - presentation mode	The subtitles are always visible in the user's field of view in the middle slightly below eye line, two-lined and each speaker has its own colour.	MUST
HUR.2.28.0	Accessibility interface for subtitles - position in viewing field	The user can select between a predefined set of positions in the viewing field (top, bottom) with a graphical user interface.	MUST
HUR.2.29.0	Accessibility interface for subtitles - size	The user can select between a predefined set of sizes (small, medium, and large) with a graphical user interface.	MUST
HUR.2.30.0	Accessibility interface for subtitles - background	The user can select between a predefined set of backgrounds (semi-transparent box, outline,-with a graphical user interface.	MUST
HUR.2.31.1	Accessibility interface subtitles - speaker location indicator	The subtitles are always positioned in the user's field of view according to personalization settings and an arrow left or right indicates the position of the current speaker.	SHOULD

HUR.2.31.2	Accessibility interface subtitles-speaker location indicator	The subtitles are always positioned in the user's field of view according to personalization settings and a-radar interactively indicates the position of the speaker by positioning a dot inside the radar field in relation to the viewer's orientation.	SHOULD
HUR.2.36.0	Switch on/off audio description	The user has the possibility to switch on/off audio description	MUST
HUR.2.37.1	Selection of Personalization options for audio description	The user has the possibility to select different languages of the service and different audio description modes (see also requirements HUR.2.40.1, HUR 2.42.1 and HUR.3.8.0)	MUST
HUR.2.38.0	Interface adapted to user device	The user interface is adapted to the device used by the user	MUST
HUR.2.40.1	Presentation mode for Audio Description	AD placed on the action (privilege of sound) - AD moves were the action is	SHOULD
HUR.2.42.1	Presentation mode for Audio Description	AD anchored to soundscape (1st person past tense) - the AD sitting next to you (left or right)	SHOULD
HUR.2.45.1	Accessibility interface for Audio Description - identify position	When there is an interesting secondary AD utterance the player places an audio beacon (beep) in that direction. Pressing the pause or "listen to beacon" button pauses the main audio and plays the AD for that object.	SHOULD
HUR.2.49.0	Accessibility interface for signer - comfort field of view	Users have the possibility to personalize the comfort field of view according to their preferences. Recommended are three levels (40%, 50%, 60%) in a 16:9 area according to the pre-pilot tests.	SHOULD
HUR.2.49.1	Accessibility interface for signer - comfort field of view	The user gets a visual feedback (dotted line) when selecting a new comfort field of view	SHOULD
HUR.2.50.0	Accessibility interface for subtitles - comfort field of view	Users have the possibility to personalize the comfort field of view according to their preferences. Recommended are three levels in a 16:9 area (50%, 60%, 70%) according to the pre-pilot tests.	SHOULD
HUR.2.50.1	Accessibility interface for subtitles - comfort field of view	The user gets a visual feedback (dotted line) when selecting a new comfort field of view	SHOULD
HUR.2.51.0	Switch on/off audio subtitling	The user has the possibility to switch on/off audio subtitling in parallel or separately to audio description.	MUST
HUR.2.52.0	Access to Audio Subtitling	The user can use a visual menu to access the service. It should be large, with a black background and white text for the highest possible contrast. Yellow will be used to highlight	MUST

		the choice of the user.	
HUR.2.53.0	Selection of Personalization options for audio subtitling	The user has the possibility to select different languages of the service	MUST
HUR.2.54.0	Volume control of interface	The user can control the volume of the main content with a graphical user interface.	MUST
HUR.2.55.0	Volume control of audio description	The user can control the volume of the AD (independently of the main volume) with a graphical user interface.	MUST
HUR.2.56.0	Volume control of audio subtitling	The user can control the volume of the AST (independently of the main volume) with a graphical user interface.	MUST
HUR.2.57.0	Personalization options for interface	The user can activate/deactivate different personalization settings with a graphical user interface.	MUST
HUR.2.58.0	Language selection for interface	The user can select the language of the graphical user interface (categories, options, icons/abbreviations for accessibility services).	MUST
HUR.2.59.0	Voice commands	The user can control all interface settings with voice commands.	MUST
HUR.2.60.0	Accessibility interface signer - language	The user has the possibility to select different languages of the signer	MUST
HUR.2.61.0	Accessibility interface - speaker location indicator	When using the radar speaker location indicator, the user has the possibility to return to the main action of the video by clicking on a specific point on the radar icon.	SHOULD
HUR.2.62.0	Progress bar	The user can monitor the progress of the video via a progress bar and can jump to a specific point in time by clicking on that point in the progress bar	MUST
HUR.2.63.0	Access to interface	The user is informed how to open the interface via a banner display once a video starts playing	MUST
HUR.3.1.0	Sign Language Service	Sign Language Service must also be considered, appearing simultaneously to the person speaking.	MUST
HUR.3.2.0	Accessibility interface for subtitles - notices for dramaturgically-significant sounds	The user gets written translations of dramaturgically significant sounds, which are important for the plot.	MUST
HUR.3.4.0	Simplified Subtitles	Simplified subtitles may be useful for users with the need of easy-to-read texts.	SHOULD

HUR.3.5.0	Immersive Subtitles information	Subtitles are always visible somewhere on the screen, whether the object they represent is visible on the screen or not.	MUST
HUR.3.6.0	Playback of 3D audio	Audio is presented as “3D audio”. This may be e.g: via a suitable surround sound speaker system (preferably including height speakers) or via a binaural signal played back via headphones	MUST
HUR.3.7.1	Different voices for main and secondary actions	The main AD track keeps playing and as the user moves their head secondary AD tracks can be played depending on the direction the user is looking. These tracks would not overlap and should use different voices for the main and secondary audio tracks.	SHOULD
HUR.3.9.0	Sign Language Service	Body shift that is traditionally used by signers to indicate a change of speaker is avoided because it makes no sense for 360° content	MUST
HUR.3.11.0	Suppression of speaker location indicator - Subtitles	If a the duration of a subtitle frame is below a given amount of time (threshold to be specified) the speaker location indicator for that frame is suppressed automatically by the player	SHOULD
HUR.3.12.0	Suppression of speaker location indicator - Signer	If a the duration of a signer segment is below a given amount of time (threshold to be specified) the speaker location indicator for that segment is suppressed automatically by the player	SHOULD

Table 1b - Key requirements for the professional user taken from D2.2 that are required for a successful implementation.

Version	Title	Description	Prioritization
PUR.1.1.0	Watch low-res preview content	The user is able to watch the preview content in low quality as flat folded view	MUST
PUR.1.1.1	Watch low-res preview content	The user is able to watch the preview content in low quality as flat unfolded view	SHOULD
PUR.1.2.0	Watch hi-res preview content	The user is able to watch the preview content in high quality as HMD view	MUST
PUR.1.3.0	Navigate preview content by angle	The user is able to watch content and to navigate around it with the help of keyboard shortcuts, scroll wheel and input fields by angle.	MUST
PUR.1.4.0	Navigate preview content by frame	The user is able to watch content and to navigate around it with the help of keyboard shortcuts, scroll wheel and input fields by frame number.	MUST

PUR.1.5.0	Navigate preview content by time	The user is able to watch content and to navigate around it with the help of keyboard shortcuts, scroll wheel and input fields by time code.	MUST
PUR.1.5.1	Navigate preview content by time	The user can add and subtract a given amount of time to the input field for timecode	MUST
PUR.1.6.0	Navigate preview content by audio	The user is able to hear 360° audio	MUST
PUR.1.7.0	File operations	The user is able to perform file operations such as import or export files (video and ImAc files)	MUST
PUR.1.7.1	File operations	The user is able to import subtitle files (STL, WebVTT (later on), EBU-TT-D (Phase 1))	MUST
PUR.1.8.0	File operations	The user is able to produce subtitle texts by (1) inserting text with keyboard, (2) symbols from a library and adding it all to the video with the following ordered steps: 1) Defining vertical and horizontal position and font size 2) Defining timecode and duration 3) Defining font colour 4) Defining viewing angle position of speaker (given as horizontal angle)	MUST
PUR.1.8.1	Add subtitle text	The user is able to create ST frames that are not related to a specific angle	MUST
PUR.1.8.2	Add subtitle text	The user is able to define the width of a "security angle" that covers a speaker. The speaker location indicators only disappear once the centre of the FOV is within the security angle. The centre of this security angle is the position of the speaker defined as the viewing angle.	SHOULD
PUR.1.8.3	Add subtitle text	The user is able to add missing spatial information to legacy subtitles, which were imported	MUST
PUR.1.8.4	Add subtitle text	The user is able to define frames in which the automatic speaker location indicator will change the field of view (this is typically done when a new scene starts or a speaker changes his position)	MUST
PUR.1.9.0	Add sign language video	The user is able to add sign language video with the following ordered steps: 1) Separating specific SL segments if necessary 2) Defining dimensions of SL video 3) Defining vertical and horizontal position 4) Defining timecode	MUST

		5) Defining position of speaker (given as horizontal angle)	
PUR.1.9.1	Add sign language video	The user is able to create SL segments that are not related to a specific angle	MUST
PUR.1.9.2	Add sign language video	The user is able to define the width of a "security angle" that covers a speaker. The speaker location indicators only disappear once the centre of the FOV is within the security angle. The centre of this security angle is the position of the speaker defined as the viewing angle.	SHOULD
PUR.1.9.3	Add sign language video	The user is able to add missing spatial information to legacy SL videos, which were imported	MUST
PUR.1.9.4	Add sign language video	The user is able to define frames in which the automatic speaker location indicator will change the field of view (this is typically done when a new scene starts or a speaker changes his position)	MUST
PUR.1.10.0	Create AD preview content	The user is able to feed a written AD script and start a text-to-speech process.	MUST
PUR.1.11.0	Add AD preview audio to video	The user is able to add a text-to-speech AD result to a video as additional audio asset.	MUST
PUR.1.12.0	Preview video and AD audio	The user is able to preview the video together with the added speech-to-text AD asset.	MUST
PUR.1.13.0	Add audio description	The user is able to add a number of simultaneous audio descriptions to different sections of the visual scene.	MUST
PUR.1.14.0	Pre-listen 3D audio content	The user is able to pre-listen immersive audio content together with immersive AD	SHOULD
PUR.1.15.0	Create a new ImAc file	It is possible to create a new accessibility file by using an existing accessibility file as template	SHOULD
PUR.1.16.0	Edit shortcuts	The user can change the shortcuts used in the editor tools for the ImAc content	SHOULD
PUR.1.17.0	Edit and preview mode for ImAc files	There are two different modes for editing and previewing ImAc content. The preview mode allows a few editing options.	MUST
PUR.1.19.0	Add subtitle text	The user can monitor the reading speed via numeric display	MUST
PUR.1.20.0	Add subtitle text	The user can monitor the number of characters per line via numeric display	MUST
PUR.1.21.0	Add subtitle text	The user has the option to activate an automatic separation of subtitle frames by a defined amount of video frames	SHOULD

PUR.1.22.0	Navigate preview content by defined number of frames	The user is able to navigate content by a customizable amount of frames for forward/backward jump	SHOULD
PUR.1.24.0	Visual display of audio content	The user can monitor the main audio content via a visual display of the sound wave	MUST
PUR.1.25.0	Add subtitle text	The user can create subtitle frames with overlapping timecode	MUST
PUR.1.26.0	Edit audio description script	The user can split and merge AD script files	MUST
PUR.1.29.0	Monitor recording of AD	The user gets a visual feedback on the start and stop of the recording of an audio file	MUST
PUR.1.30.0	Define fading level of main audio	The user can choose between several levels of fading for the main audio	MUST
PUR.3.1.0	Accessing content for ImAc enrichment	The user uses a GUI for accessing omnidirectional (video) and ImAc content files (ST, SL video, AD) for selecting and uploading/downloading files.	MUST
PUR.3.2.0	Checking content for ImAc enrichment	The user is able to check ImAc media regarding: file name, file size, content type, integrity, assignment	SHOULD
PUR.3.3.0	Assigning content for ImAc enrichment	The user is able to assign ImAc files to omnidirectional media files	MUST
PUR.3.4.0	Triggering content packaging and distribution	The user is able to trigger and monitor the packaging of ST, SL and AD enhanced media items	MUST
PUR.3.5.0	Checking state of content packaging and distribution	The user is able to check the state of packaging of enhanced media items.	SHOULD
PUR.3.6.0	Locally retrieving the packaging result	The user is able to direct the packaged ImAc result as local retrieval.	MUST
PUR.3.7.0	Directing the packaging result to a different resource	The user is able to direct the packaged ImAc result forwarded to a remote resource.	MUST

PUR.3.8.0	Configure the signalization of ImAc services	The user is able to configure signalization of ImAc content for distribution and playback.	SHOULD
PUR.3.9.0	Monitor the signalization of ImAc services	The user is able to monitor signalization of ImAc content for distribution and playback.	SHOULD
PUR.3.10.0	Defining speaker location indicator options	The user is able to define the speaker location indicator options which are offered to the home user	SHOULD
PUR.3.11.0	Speaker introduction	Each speaker is introduced in the subtitles (e.g. by name or "man"/"woman") when speaking for the first time	MUST
PUR.3.12.0	Web interface for high-resolution file upload	The user can access a simple web interface to provide metadata about high resolution content that he/she uploaded to the SFTP. This metadata will trigger the automatic generation of low-res content on ACM.	SHOULD
PUR.3.13.0	Assign users to edit ImAc files	The user can assign one or more subtitlers at a time to edit ImAc files.	MUST
PUR.3.14.0	Define thumbnail for display of omnidirectional media	The user can define the thumbnail shown in the GUI as preview for an omnidirectional media file in the list of files.	SHOULD
PUR.3.15.0	Export AD script as text file	The user is able to export an AD script as text file	SHOULD

2.2. Web based design

The ImAc project has focused development on building web-based components. This includes the player, which can run in a web browser and the authoring tools are provided in a cloud-based infrastructure.

This brings many advantages to the development cycle. Building web-based applications is relatively quick in terms of development cycle and easy to deploy and test. Being web based also makes the player very accessible to end-users. Anyone with a web browser and Internet connection can open the player and consume the content. It also allows the professional users to work on authoring content from home.

The ImAc project is focused only on 360° video, which can be achieved with WebVR technologies. There is no requirement to use heavier technologies such as Unity to render the graphics as we are only playing videos rather than interacting with 3d objects in real time.

2.3. Considerations for an end to end subtitle workflow

This chapter describes the overall approach for ImAc's subtitle service from a technical perspective. Based on the initial user requirements that were collected from the focus group tests, this strategy was drafted to provide a technical basis for the service that considers all aspects of the subtitle workflow. Since modifications of the subtitle service can be expected during upcoming tests and pilot phases, the system provides flexibility especially regarding changes of subtitle presentation options.

The chapter is divided into two sections:

- A general part, which includes specifications that have implications on the overall system
- A content related part, which mainly describes considerations for the different requirements on the behaviour and presentation of subtitles

2.3.1 General specifications

Table 2 – List of relevant links regarding subtitle service

Description	**Link**
IMSC	https://www.w3.org/TR/ttml-imsc1.0.1/
TTML1	https://www.w3.org/TR/ttml1/
TTML2	https://www.w3.org/TR/2018/CR-ttml2-20180313/
EBU-TT	https://tech.ebu.ch/subtitling
WebVTT	https://www.w3.org/TR/webvtt1/
EBU STL	https://tech.ebu.ch/docs/tech/tech3264.pdf
DVB BlueBook A174	https://www.dvb.org/resources/public/standards/a174_dvb_ttml_subtitling_systems.pdf
HbbTV 2.0	http://www.hbbtv.org/resource-library/#specifications
Overview of various TTML profiles by W3C	https://www.w3.org/AudioVideo/TT/docs/TTML-Profiles.html
Imsc.js	https://github.com/sandflow/imscJS

2.3.1.1 Subtitle Distribution Format

ImAc's subtitle format for distribution is a subset of the TTML (Timed Text Mark-up Language). The TTML profile chosen in ImAc is IMSC (Internet Media Subtitles and Captions) text profile.

The only relevant alternative to TTML format(s) is WebVTT (Web Video Text Tracks Format), a W3C working draft that specifies style, position and time mark-up for HTML text tracks. This document does not claim to provide a detailed comparison between TTML and IMSC. In ImAc TTML was chosen, mainly because:

- TTML and especially some subsets (EBU-TT, SDP-US) were specified to fit for professional use (e.g. by broadcasters). WebVTT on the other side is web related and not suitable for other areas of the subtitle workflow.
- TTML serves all areas of subtitle workflows, from production to distribution. It includes mark-up information and metadata to feed all distribution
- There is a very active community pushing TTML (especially some subsets). This includes stakeholders from the industry, content providers and standardization groups.

However, it has to be stated that WebVTT is widely supported by browsers today already. TTML (and IMSC) still lacks native browser support.

Various profiles and subsets of TTML exist, some are closely related and only differ in a few aspects. Other TTML subsets are for example: EBU-TT, EBU-TT-D, EBU-TT Part 3 (Live), SDP-US, CFF-TT, SMPTE-TT.

IMSC was chosen for various reasons:

- IMSC is the TTML profile that draws the most attention by standardization bodies and the industry right now.
- It can be extended with additional metadata and extensions.
- It is compliant with the current Netflix requirements.
- In 2017, the DVB referenced IMSC for their TTML Subtitling Systems (BlueBook A174).
- Since it is (almost) a superset of EBU-TT-D, an IMSC decoder will play EBU-TT-D streams. EBU-TT-D is referenced in the DASH profile for HbbTV2.0 for example.
- It's likely that in near/mid-term future, there will be a wider support for IMSC in the market.

Note: In the following, both formats TTML and IMSC are referred to. When talking about general aspects, that are the same for TTML and all its subsets, then TTML is used. When talking about ImAc's subtitle format, IMSC is used. However, there is no hard line in this rule and usage may not always be consistent.

One aspect of the subtitle service is streaming capabilities. Basic requirements in ImAc are similar to traditional subtitle streaming, but shall be mentioned here nevertheless:

- It must be possible to stream subtitles within common streaming formats.
- It must be possible to stream and identify multiple subtitle streams with different languages for one program.

The IMSC format can be streamed within the MPEG-DASH streaming format, which is used in ImAc – see chapter 3. As basis for the implementation, ImAc refers to the specification for transportation of EBU-TT-D in HbbTV 2.0. In HbbTV 2.0, two options are described: in-band delivery and out of band delivery. Both are options in ImAc, although the out of band solution might be the simplest way to realize first implementations. In the framework of ImAc, multiple subtitles might be also transmitted for one program into the MPEG-DASH stream(s). Synchronization capabilities of multiple subtitles will be proposed (and will be signalized accordingly).

2.3.1.2 Subtitle Production Format

It is important for the ImAc architecture to specify the subtitle distribution format. And as indicated above, the IMSC format was chosen to support an easy integration into production workflows. Nevertheless, other formats WILL be used during the production process, which would then be transformed to IMSC for play-out. Production and distribution use different formats although TTML attempts to unify the technology by providing a production alternative of distribution EBU-TT-D as EBU-TT. This format is based on Teletext and the production format in most broadcaster environments.

In ImAc, subtitle editing will be done using IMSC directly, but the EBU STL format will be supported as an import format to ensure that traditionally produced subtitle files can be used for the ImAc services.

2.3.1.3 Management of subtitle files

The management of subtitle files within the premises of a service provider is an important task in order to provide a reliable subtitle service (to professional and end users). But the workflows of different providers vary and depend very much on the overall system environment of a company. Thus, it is not in the scope of ImAc to draft a generic architecture for data handling.

In most cases, such workflows are independent of the content of a file, except for some metadata that ensures correct processing of the files. Example: The information of the subtitle language is required for a correct signalization of the subtitle stream.

However, ImAc installs an exemplary workflow for accessibility data. The Content Manager for accessibility data demonstrates how management of accessibility data can be smoothly included in production and distribution workflows. More details can be found in Chapter 3.

2.3.2 Content related considerations

2.3.2.1 Positioning of subtitles

Following the collected user requirements, a new extension for subtitle formatting will need to be added in ImAc: spatial information. In traditional (2D) media, subtitles are positioned as well, but for the concepts that are envisioned in ImAc, positioning plays a completely different role. Some essential differences arise when the video is mapped to a projection plane and only a part of the video is rendered: In 360°, subtitles and video are not linked as they are with "normal" video. The subtitle will never be distorted, it will not change size due to zooming, it will never be out of the Field of View (FOV) and a subtitle will not look like it is part of the scene.

The final rendering position of a subtitle may be calculated only on the user device where the information on the current viewing direction is available. That means the player needs to implement additional processing for correct subtitle rendering.

For the following considerations, some characteristics of a 360° environment are defined (these definitions comply with the ImAc system):

- The video is stored as a 2D image that is intended for an equirectangular projection.

- For presentation, this 2D image is mapped (projected) to a sphere. Coordinates on the sphere can be expressed by latitude and longitude.
- The viewer will see a part of this sphere that is projected (back) to a flat screen. This is called “field of view” (FOV).
- Within the FOV, the subtitles are placed within a safe area. That means the subtitles will never be placed at the very edge of the field of view (as it is done in traditional TV as well).

Typically, the subtitle-rendering plane (or “root container” in TTML) is equal to the full width and height of the underlying video. But it is not possible to simply follow that practice in ImAc, since the FOV (where subtitles shall be rendered on) is a) smaller than the video size; and b) not at a fixed (video) position. The TTML root container for the 360° environment could be defined in two different ways:

1. The rendering plane is defined as the FOV.
2. The rendering plane is defined as the stored 2D video, following the same equirectangular projection mapping (i.e. a location on the 2D rendering plane can be mapped to a point on the sphere).

In each approach the subtitles have to be processed for final rendering. Processing regarding icons that indicate directions (arrows, radar, etc.) will be similar for both options, only the information for speaker position may come from different attributes.

Speaker positions will be added by user-defined extensions, which is allowed by TTML and IMSC (as foreign namespace extensions). Introducing these extensions into TTML is the preferred way in ImAc in order to push standardized solutions. The speaker position information will be used by the player to add icons (arrows, radar, etc.) that point to the speaker or to re-position the subtitles in the FOV in relation to the current speaker position.

2.3.2.2 User preferences

This section will not repeat what user preferences will be available in ImAc (D2.3) but describe how they might be realized in the system. Three scenarios are possible:

- Transmit user preferences with dedicated extensions (e.g. font sizes in the subtitle format). The player makes preferences available to the user in a way that is described in the content stream.
- Handle user preferences in the player. In such a case, the content stream may include no metadata regarding preferences.
- Mixed approach: Some user preferences are indicated in the stream (e.g. what presentation modes should be offered), others are handled in the player only (e.g. font sizes).

It is a matter of sharing rendering responsibility between player application and content provider, and there will be no sharp line here, because a player may overwrite preferences of the content stream. For some preferences, it makes sense to have at least an “editor’s choice” signalled within the stream.

In the first pilot phase, all user preferences will be realized within the player application, because it is the easiest way to realize early pilots in ImAc. In the second phase, it might be considered to move some of the information for user preferences to the content stream.

TTML can easily be enriched with user extensions, thus the subtitle format will not be a limiting factor. However, finding a standardized way for example for indicating available font sizes is preferred in ImAc.

2.3.2.3 Depth

The focus of ImAc is on environments with all elements of a scene (like media content and user interface elements) having the same depth. However, real 3D environments shall also be addressed in the project. For subtitles a few issues need to be investigated here, but this document will not provide a strategy for presentation of subtitles in 3D environments.

User requirements in this field were not collected yet in the project. Still, a basic assumption can be made: That subtitles should be rendered on top of (all) other objects, such that the text is not obscured by anything. The background for this assumption: ImAc’s focus group questionnaires have revealed that users prefer subtitles to be always visible in the FOV. Apparently, users want to able to read them any time, which is comprehensible. The same demand can be assumed for 3D, which means the subtitle text must be visible at all time. This assumption leads to the following question:

How can subtitles be presented in a comfortable way, when the scene contains elements close to the viewer?

When subtitles are placed as another element into the scene and when a comfortable depth for that element is chosen for that subtitle, that means that no element can be placed closer to the viewer than the subtitle is (at least not at the same location). Adding subtitles in a real 3D environment will prOBAbly demand for compromises one way or another.

TTML supports basic options for specifying depth information on subtitles. The options available are likely to be sufficient to indicate at what depth the subtitle should be rendered.

2.3.2.4 Safe area

For television systems the EBU R95 defines a graphic save area of 90% of the active picture. In TTML, the safe area is typically handled by the size (and position) of the region element. In 360°, the safe area could be indicated the same way, but only when the TTML root container is defined to have the same size as the FOV (which will be the case in IMAC). An alternative option is that the player handles the safe area by addressing a corresponding rendering plane. This plane should then be fully used (without leaving margins, because the player already considered safe area). Here, the editor would have no possibility to indicate the save area (at least not with native TTML features).

For the first pilot phase, ImAc has handled area only in the player. Region sizes from the TTML document will be ignored. Depending on standardization activities, this may change in the second pilot phase.

Another aspect that is related to save area is the aspect ratio of the FOV that is unknown to the editor. Thus, the editor might want to suggest a target aspect ratio. This has not been done in the first pilot phase, but might be introduced later in the project.

2.4. Considerations for an end to end AD and subtitle workflow

This chapter describes the overall approach for ImAc's Audio Description (AD) service from a technical perspective. Based on the initial user requirements that were collected from the focus group tests, this strategy was drafted to provide a technical basis for the AD service.

One result of the ImAc focus group tests shall be mentioned here explicitly: It seems that a major contribution to the user experience may come from an appropriate main audio mix rather than the AD service itself.

Thus, this chapter not only describes the AD service, but includes aspects for audio in general and gives an overview of the influence of various audio formats and playback systems on the immersive (audio) experience.

2.4.1 Audio Systems / Audio Distribution Formats

All audio content (main audio and AD) will be provided in different audio formats in order to support a wide range of user devices. The following audio systems will be supported:

- Stereo: The stereo system will be supported
- Ambisonics: This is a client-side rendering system, which generates binaural audio for headphones. First Order Ambisonics is easy to achieve but does not provide a fine-grained immersion. Higher Order Ambisonics results in a good spatial resolution, but has higher costs in hardware requirements on client side.
- Binaural: The 2-channel format for headphones, which allows good spatial resolution. Since this format is already rendered on production side, the audio is static and does not support dynamic changes by head/display rotations. In comparison to the stereo signal, the user may realize a different "sound colour".

It was considered to deliver object-based audio up to the end user device. This allows for high flexibility regarding the audio scene as well as the playback system, because the audio is rendered on the user device. However, relatively high processing power is needed to perform the client-side rendering. Additionally, the format for delivery of object-based audio has not been established yet. Thus, it was decided to rule out this option for this project.

It is planned to use the common audio codec AAC for delivery of all audio streams.

2.4.2 Audio Production Formats

On the production side, object based audio is used preferably: It allows the generation of any output format on demand. Since all audio information is available as "raw data", the requirements for rendering are relatively high as well as the data traffic. The object based audio scene needs to be rendered into target output formats, which are defined by the producer or within the play-out process.

AD can be edited within the object based audio editor (OBA editor) as any other audio object, or – as it is planned in ImAc – externally within an editor specifically addressed for AD productions. In the latter case, the AD track must be imported into an existing audio scene.

The OBA editor provides an interface for importing AD tracks including their metadata (e.g. regarding positioning and gain).

Stereo and 5.1 productions can be represented by object-based audio as well. For instance, a stereo production would be imported into an audio scene by adding two audio objects – one for each channel. These objects are placed at the location where the loudspeakers would stand in a typical stereo setup. The same approach can be used for 5.1 or any other channel based production.

That way, all typical production formats can be supported in ImAc. This is relevant, since object based audio productions are still rare.

2.4.3 Management of AD file workflow

The management of AD files (and audio files in general) within the premises of a service provider is an important task in order to provide a reliable AD service (to both professional and end users). But the workflows of different providers vary and depend very much on the overall system environment of a company. Thus, it is not in the scope of ImAc to draft a generic architecture for data handling.

However, ImAc installs an exemplary workflow for accessibility data. The Content Manager for accessibility data demonstrates how management of accessibility data can be smoothly included in production and distribution workflows. More details can be found in chapter 3.

The major task here is to add (or mix) AD to the main audio. That may be done automatically outside the editor software. In ImAc, this step is realized by the “AD import interface” of the OBA editor that supports all functionalities to embed AD into the audio scene.

After all output formats have been rendered, the AD service needs to be signalized in MPEG DASH such that the player can identify the AD services in the stream.

The AD service in general as well as its language will be signalized as defined by the DASH standard and DVB specification XXX (). The DVB specification defines the usage of the element “Accessibility” in the DASH-MPD.

Signalization of ImAc specific features, two custom attributes have been added to the element “Representation” in the DASH-MPD:

1) ad-mode – The attribute describes for each AD track which mode is used.
2) ad-volume – This attribute describes for each AD track the volume level of the AD speaker in relation to the main audio.

Refer to section “User Preferences” below for further details on these features.

2.4.4 Immersive sound and dependencies from playback systems

This section shortly describes the influence of different playback systems on the immersive experience. Only a rough overview can be given here, but it is important to understand that some of the collected user requirements lead to preconditions regarding user hardware and audio formats.

The way that the audio scene should act also depends on the viewing environment that is used. Basically, there are two different kinds of display devices, that will be called “fixed” and “non-fixed”:

- Fixed display device means that the display device is not rotating or moving. In order to change the viewing direction, another part of the video will be moved into the field of view (FOV). One could say that the video rotates around the user. The user doesn't rotate but always looks in the same direction. Navigation is often done by mouse, arrow keys, swiping on touch display.
- Non-fixed display device means that the user needs to look around (move his/her head) in order to change the FOV. One could say that the video does not rotate around the user, but the user will turn to watch different areas of the 360° scene.

Note: Tablets can be used as both fixed and non-fixed displays, since the players often allow two options to change the FOV:

- *Swiping over display (tablet acts like a fixed display)*
- *Using the internal sensors when user turns around with the tablet in hand (tablet acts like a non-fixed display)*

It is assumed that the represented audio scene should align with the video for the best immersive experience. As a result, the kind of display used defines if the audio scene must rotate around the user or not.

Table 3 provides an overview of functionalities and behaviours of different viewing and listening setups.

Table 3 - The functionalities and behaviours of different AD setups

#	Display	Speaker	Notes
1	Fixed display (e.g. PC+browser)	Loudspeaker setup (Stereo, 5.1, …)	Video scene: rotating Playing Ambisonic: The audio scene will rotate together with the video. Playing Stereo or 5.1: The audio scene will not rotate together with the video.

2	Fixed display (e.g. PC+browser)	Headphones	Video scene: rotating Playing Ambisonic: The audio scene will rotate together with the video when the user doesn’t turn his head. Additional head movements will not be considered by default. Playing Stereo: The audio scene will not rotate together with the video. Head movements will not be considered by default. Playing binaural sound (no head tracking): Same as “Stereo”. Playing binaural sound (with head tracking and corresponding rendering): When head movements and FOV changes can be processed, the audio scene can be reproduced correctly, that means both FOV changes and head movements are considered. The audio scene rotates together with the video even when the user turns his/her head.
3	Non-fixed display (e.g. HMD or Tablet+Sensor)	Loudspeaker setup (Stereo, 5.1, …)	Video scene: not rotating This is a rather unusual scenario, except for one use case: A tablet is used, its internal sensor changes the FOV and audio is played via internal (mono or stereo) speakers. But in this case, the tablet speaker moves as well, which is different from a usual speaker setup. An immersive audio experience is not achieved by this setup and this use case will not be explored in ImAc.

4	Non-fixed display (e.g. HMD or Tablet+Sensor)	Headphones	Video scene: not rotating This is the main viewing environment for ImAc and will lead to the best immersive experience. Head tracking data will be available since it is used to change the FOV. Playing Ambisonic: The audio scene will rotate together with the video. Additional head movements will be considered as well. Playing Stereo: The audio scene will rotate with head movements and will then diverge from the video scene. Playing binaural sound (no head tracking): Same as “Stereo”. Playing binaural sound (with head tracking and corresponding rendering): Same as “Ambisonic” but better localization of audio sources can be achieved.

A binaural stream only represents one viewing direction. To get the best immersive impression, the binaural audio needs to be adapted when viewing direction changes. As a precondition, the audio processor must know the viewing angle. Basically, there are two approaches to reproduce binaural audio for the current viewing direction:

- A stream for each viewing direction is pre-rendered and provided (typical resolution: 1°)
- The binaural signal is rendered dynamically

Note: The delay between head movement and corresponding change of the binaural signal must be small (around 40ms). Otherwise the impression of a stable audio scene will break.

The different ImAc formats that are supported were chosen to provide acceptable audio for all viewing environments. In general, playing binaural audio on headphones, where the binaural sound changes according to the user’s head movements, will deliver the best user experience. The support of this feature is targeted for the second pilot phase.

2.4.5 User Preferences

The following user preferences regarding AD presentation will be supported in ImAc:

- The user can choose from different gain levels for the AD channel (in relation to main mix).
- The user can choose from different locations for the AD speaker (i.e. from which direction the sound will come from).

There are two options regarding the realization of this feature that were discussed in ImAc:

1. Deliver the AD channel as a separate stream and mix it with the main audio on the user device.
2. Pre-render all variations a user can choose from and provide several streams.

Client-side mixing allows for more flexibility and is definitely the way to go when object based audio is brought up to the client device. Since in ImAc object based audio is only used at server side (at least for the first pilot phase), the different user preferences will be pre-mixed by the service provider and exposed to the client within the distribution (e.g. MPEG-DASH manifest).

2.5. Considerations for an end to end SL workflow

This final chapter describes the overall approach for ImAc's Sign Language. Based on the initial user requirements that were collected from the focus group tests, this strategy was drafted to provide a basis for the service that considers all aspects of the sign language workflow. A specific web based sign language editor in implemented for producing sign language for 360º media.

Main features:

- Web based for universal access.
- Uses the webcam of your choice to record the video segments
- Video control buttons equipped with shortcuts for convenience (play, pause, step forward and backward, fast forward and backward, frame by frame, jump configurable number of frames)
- Possibility to work with different digital video codecs (H264, VP8, VP9), formats (mp4, webm) and frame rates (i.e. 25fps, 24fps, 30fps etc.)
- Write the script of the speaker and/or possible comments on the segments in the area predicted for such action, and also define each segment IN and OUT time codes, as well as a suitable angle.
- Shortcuts for the most important actions.
- Navigation through the segments (going backward or forward, jumping to first or last line, choosing a subtitle by entering its number)
- Navigation through video via its angle.
- Possibility of seting a segment with no angle, in case there is no present character.
- Segments are editable, as they can be removed or added and their sequence can be modified easily
- Different modes of preview depending on the functionality:
    - Edit: during the production of the segments.
    - Forced preview: used as a mode of verification. In this mode the segments and angle are bound with the video. Signer cannot move freely in the video under any angle as the video itself takes you to the speaker angle.
    - Free preview: this mode is used for verification as well, the difference with forced view is that as if it is playing with an HMD. Segments are bound, angle is not. It

    means that the user can navigate in the video independently from the speaker angle.

- Creation and editing of video segments
    - Video segment recording. A countdown during the recording is provided to show the user how much time is left according to the video segment time codes
    - “long test” and “short test” are available options in order to perform 5 and 2 seconds test respectively before the TC In of the video segment
- Segments content is editable by corresponding button, the button takes the user to a window to do so:

  The editing takes places in two modes:

    - Split: Split the segment into different ones, these shorter segments can be added or removed
    - Cut: Shorten the video segment, either from the beginning or the end.
- Possibility to auto-save the work while working on it
- Full script and their time codes on the right of the screen and jump to the desired segment by clicking on it.

3. ARCHITECTURAL REPRESENTATION

This document presents the architecture as a series of views, based on the '4+1' View Model of software architecture [1].

A set of scenarios provide a description of the use-case view of the software architecture describing different situations and/or use cases that represent some significant, central functionality.

The logical view describes the most important classes, their organization in service packages and subsystems, and the organization of these subsystems into layers.

The process view captures the concurrency and synchronization aspects of the design, and describes the tasks (processes and threads) involved in the system's execution, their interactions and configurations. It also describes the allocation of objects and classes to tasks.

The deployment view of the architecture describes the various physical nodes for the most typical platform configurations. It also describes the allocation of tasks (from the Process View) to the physical nodes.

Based on the '4+1' model we do not provide a Development View as the development team structure and software management is discussed in the project proposal. Figure 2 demonstrates the structure of the ImAc architectural representation.

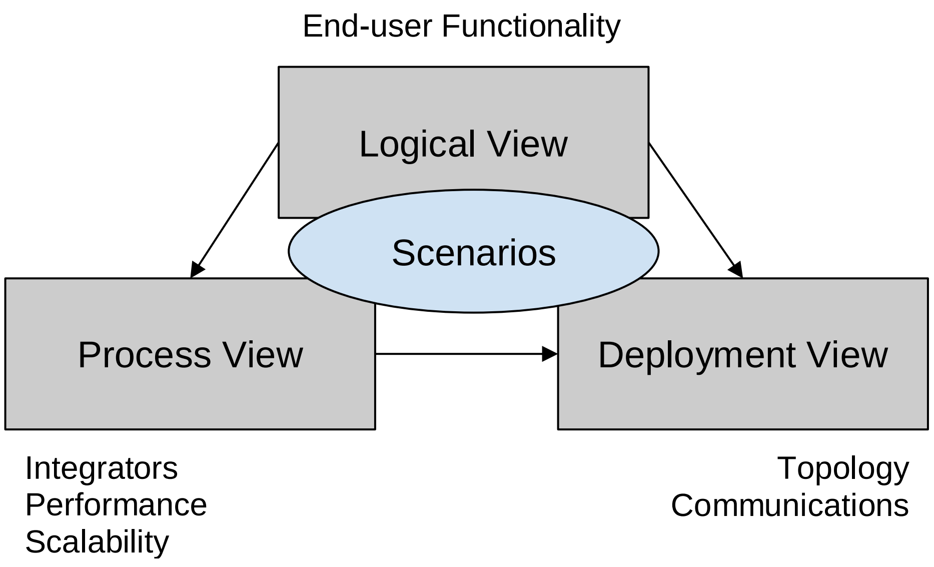


Figure 2 - The structure of the ImAc architectural representation

3.1. Scenarios

Different scenarios are described using a use-case view of the software architecture in order to highlight significant and central functionality. It also describes the set of scenarios and/or use cases that have a substantial architectural coverage (that exercise many architectural elements) or that stress or illustrate a specific, delicate point of the architecture. Table 4 shows an overview of the scenarios defined within the project.

Table 4 - Overview of Scenarios

Production Editors	S1	Production of New Subtitles
	S2	Subtitle Verification and correction
	S3	Production of Audio Description / Audio Subtitles
	S4	Audio Description Verification and Correction
	S5	Production of Sign Language
	S6	Sign Language Verification and Correction
	S7	Object Based Audio Editing
Accessibility Content Manager	S8	Assignment of videos for the accessibility content production
	S9	Getting an Accessibility Content file
	S10	Generating Different audio formats with an audio renderer
Content Packaging and Distribution	S11	Preparation of Contents
	S12	Distribution
Player	S13	Consumption of media contents

3.1.1 Production Editors

In all the following use-cases the user is a professional user and the requirements that each scenario satisfy are listed.

3.1.1.1 New Subtitling

Table 5 – Scenario for production of new subtitles

ID	Description	Requirements
S1.1	User accesses the production web page via the web browser.	
S1.2	User enters username and password.	
S1.3	A window with the list of videos to be subtitled by the user appears.	
S1.4	User either creates a new video asset, opens an existing one or imports an external file to be subtitled and presses the edit button.	PUR.1.7.0 PUR.1.7.1 PUR.1.15.0
S1.5	The web-subtitling editor opens with the video in the preview window and with no subtitles.	PUR.1.1.0 PUR.1.1.1
S1.6	User creates a new subtitle with all the associated metadata (timecode, character, angle of view, etc.). The user does this using the video player buttons and mouse to move around the video in order to find the entry and exit points of the subtitle, find the character (by video panning) and listen to the corresponding character.	PUR.1.8.0 PUR.1.8.1 PUR.1.8.2 PUR.1.8.3 PUR.1.8.4 PUR.1.25.0
S1.7	The subtitle is simulated over the preview video window.	PUR.1.3.0 PUR.1.4.0 PUR.1.5.0 PUR.1.5.1
S1.8	User moves to the next empty subtitle by pressing the corresponding key.	
S1.9	User repeats the same process with the rest of the subtitles.	

S1.10	User checks the subtitling restrictions by pressing the corresponding button.	PUR.1.19.0 PUR.1.20.0 PUR.1.21.0
S1.11	If a subtitle doesn't comply with the restrictions, a message appears for the user to correct it before continuing with the rest of the checking process.	
S1.12	User corrects the subtitle to comply with the restrictions.	
S1.13	User repeats the same process until all subtitle items are compliant. In this last case, the simulation button is pressed.	
S1.14	User presses the simulation button to verify the final result.	
S1.15	User uses the preview video player buttons to move around and see the result accordingly. For instance if the user presses the play button the subtitles will be displayed and deleted over the preview video in their respective times along with the video playback so as to simulate the final result.	PUR.1.2.0 PUR.1.3.0 PUR.1.4.0 PUR.1.5.0 PUR.1.5.1 PUR.1.6.0 PUR.1.22.0 PUR.3.11.0
S1.16	User can also press the HMD simulation by pressing the corresponding button.	
S1.17	User puts on the HMD to watch the final result.	
S1.18	User can move up and down the subtitles if specific ones need to be corrected.	
S1.19	The subtitling is saved automatically in the Accessibility Content Manager	PUR.1.7.0

3.1.1.2 Subtitle verification and correction

Table 6 - Scenario for subtitle verification and correction

ID	Description	Requirements
S2.1	User accesses the production web page via the web browser.	
S2.2	User enters username and password.	
S2.3	A window with the list of videos to be verified by the user appears.	
S2.4	User opens an existing video to be verified and presses the edit button.	PUR.1.7.0
S2.5	The web-subtitling editor opens with the video in the preview window and with the subtitling.	PUR.1.1.0 PUR.1.1.1 PUR.1.17.0
S2.6	User checks the subtitling restrictions by pressing the corresponding button.	
S2.7	If a subtitle doesn't comply with the subtitling restrictions a message appears for the user to correct it before continuing with the rest of the checking process.	
S2.8	User corrects the subtitle to comply with the subtitling restrictions.	
S2.9	User repeats the same process until there's no subtitle that does not comply. In this last case the simulation button activates.	

S2.10	User presses the simulation button to verify the final result.	
S2.11	User uses the preview video player buttons to move around and see the result accordingly. For instance if the user presses the play button the subtitles will be shown and deleted over the preview video in their respective times along with the video playback so as to simulate the final result.	
S2.12	If the subtitling and the video are not in sync, user presses the offset button and enters the right entry time (TCin) that is required for the current subtitle. The same offset is applied to the rest of subtitles times.	
S2.13	User presses the simulation button again to verify the final result.	
S2.14	User can also press the HMD simulation by pressing the corresponding button.	PUR.1.2.0 PUR.1.3.0 PUR.1.4.0 PUR.1.5.0 PUR.1.5.1 PUR.1.6.0
S2.15	User puts on the HMD to watch the final result.	PUR.1.2.0
S2.16	User can move up and down the subtitles if specific ones need to be corrected.	
S2.17	The subtitling is updated automatically in the Accessibility Content Manager database.	PUR.1.7.0

3.1.1.3 New Audio Description

Table 7 - Scenario for production of Audio Description

ID	Description	Requirements
S3.1	User accesses the production web page via the web browser.	
S3.2	User enters username and password.	
S3.3	A window with the list of videos to be audio described by the user appears.	
S3.4	User creates a new video asset, opens an existing file or imports an external file to be audio described and presses the edit button.	PUR.1.7.0 PUR.1.15.0
S3.5	The web audio description editor opens with the video in the preview window and with no audio description.	PUR.1.1.0 PUR.1.1.1
S3.6	User creates a new audio description segment with all the associated metadata (TCs, character, angle of view, etc.). The user can use the video player buttons and mouse to move around the video in order to find the entry and exit points of the audio description, find the character (by video panning) and watching the corresponding scene.	PUR.1.10.0 PUR.1.11.0 PUR.1.12.0 PUR.1.13.0 PUR.1.26.0 PUR.1.30.0
S3.7	User moves to the next empty audio description segment by pressing the corresponding key.	
S3.8	User repeats the same process with the rest of the audio description segments.	

S3.9	User checks the audio description restrictions by pressing the corresponding button.	
S3.10	If an audio description segment doesn't comply with the restrictions a message appears for the user to correct it before continuing with the rest of the checking process.	PUR.1.29.0
S3.11	User corrects the audio description segment to comply with the restrictions.	
S3.12	User repeats the same process until there's no audio description segment that does not comply. In this last case the simulation button activates.	
S3.13	User presses the simulation button to verify the final result.	PUR.1.3.0 PUR.1.4.0 PUR.1.5.0 PUR.1.5.1
S3.14	User uses the preview video player buttons to move around and see the result accordingly. For instance if the user presses the play button the audio description segments will be playback mixed with the preview video audio in their respective times along with the video playback so as to simulate the final result.	PUR.1.14.0 PUR.1.22.0 PUR.1.24.0 PUR.1.6.0
S3.15	User can also press the HMD simulation by pressing the corresponding button.	
S3.16	User puts on the HMD device to watch the final result.	PUR.1.2.0
S3.17	User can move up and down the audio description segment if specific ones need to be corrected.	
S3.18	The audio description is saved automatically in the Accessibility Content Manager database.	PUR.1.7.0
S3.19	User might export a json file for description of AD position/gain for the OBA Editor together with raw AD audio files.	

3.1.1.4 Audio Description Verification and Correction

Table 8 – Scenario for Audio Description Verification and Correction

ID	Description	Requirements
S4.1	User accesses the production web page via the web browser.	
S4.2	User enters username and password.	
S4.3	A window with the list of videos to be verified by the user appears.	
S4.4	User opens an existing video and presses the edit button.	PUR.1.7.0
S4.5	The web editor opens with the video in the preview window and with the audio description.	PUR.1.1.0 PUR.1.1.1 PUR.1.17.0
S4.6	User checks the audio descriptions restrictions by pressing the corresponding button.	
S4.7	If an audio description doesn't comply with the audio description restrictions a message appears for the user to correct it before continuing with the rest of the checking process.	

S4.8	User corrects the audio description to comply with the audio description restrictions.	
S4.9	User repeats the same process until there's no audio description that does not comply. In this last case the simulation button activates.	
S4.10	User presses the simulation button to verify the final result.	
S4.11	User uses the preview video player buttons to move around and see the result accordingly. For instance, if the user presses the play button the audio description will be played and deleted over the preview video in their respective times along with the video playback so as to simulate the final result.	
S4.12	If the audio description and the video are not in sync, user presses the offset button and enters the right entry time (TCin) that is required for the current audio description. The same offset is applied to the rest of audio description times.	
S4.13	User presses the simulation button again to verify the final result.	
S4.14	User can also press the HMD simulation by pressing the corresponding button.	
S4.15	User puts on the HMD device to watch the final result.	
S4.16	User can move up and down the subtitles if specific ones need to be corrected.	
S4.17	The audio description is updated automatically in the Accessibility Content Manager database.	PUR.1.7.0 PUR.3.15.0

3.1.1.5 New Sign Language

Table 9 - Scenario for Production of new sign language

ID	Description	Requirements
S5.1	User accesses the production web page via the web browser.	
S5.2	User enters username and password.	
S5.3	A window with the list of videos to be signed by the user appears.	
S5.4	User creates a new video asset, opens an existing file or imports an external file to be signed and presses the edit button.	PUR.1.7.0 PUR.1.15.0
S5.5	The web sign language editor opens with the video in the preview window and with no sign language.	PUR.1.1.0 PUR.1.1.1
S5.6	User creates a new interpreter segment with all the associated metadata (TCs, character, angle of view, etc.). For that user can use the video player buttons and mouse to move around the video in order to find the entry and exit points of the subtitle, find the character (by video panning) and listen to the corresponding character.	PUR.1.9.0 PUR.1.9.1 PUR.1.9.2 PUR.1.9.3 PUR.1.9.4
S5.7	The interpreter video is shown in an independent window next to the preview video player window.	PUR.1.3.0 PUR.1.4.0 PUR.1.5.0

		PUR.1.5.1
S5.8	User moves to the next empty interpreter segment by pressing the corresponding key	
S5.9	User repeats the same process with the rest of the interpreter segments.	
S5.10	User checks the sign language restrictions by pressing the corresponding button.	
S5.11	If an interpreter segment doesn't comply with the restrictions, a message appears for the user to correct it before continuing with the rest of the checking process.	
S5.12	User corrects the interpreter segment to comply with the restrictions.	
S5.13	User repeats the same process until there's no interpreter segment that does not comply. In this last case the simulation button activates.	
S5.14	User presses the simulation button to verify the final result.	
S5.15	User uses the preview video player buttons to move around and see the result accordingly. For instance if the user presses the play button the interpreter video will be played back in their respective times in its window next to the video playback so as to simulate the final result.	PUR.1.22.0
S5.16	User can also press the HMD simulation by pressing the corresponding button.	
S5.17	User puts on the HMD device to watch the final result.	PUR.1.2.0
S5.18	User can move up and down the interpreter segment if specific ones need to be corrected.	
S5.19	The sign language is saved automatically in the Accessibility Content Manager database.	PUR.1.7.0

3.1.1.6 Sign Language Verification and Correction

Table 10 - Scenario for sign language verification and correction

ID	Description	Requirements
S6.1	User accesses the production web page via the web browser.	
S6.2	User enters username and password.	
S6.3	A window with the list of videos to be verified by the user appears.	
S6.4	User opens an existing file to be verified and presses the edit button.	PUR.1.7.0 PUR.1.15.0
S6.5	The web editor opens with the video in the preview window and with the subtitling.	PUR.1.1.0 PUR.1.1.1 PUR.1.17.0
S6.6	User checks the sign language restrictions by pressing the corresponding button.	
S6.7	If a sign language doesn't comply with the sign language restrictions a message appears for the user to correct it before continuing with the rest of the checking process.	
S6.8	User corrects the sign language to comply with the sign language restrictions.	

S6.9	User repeats the same process until there's no sign language that does not comply. In this last case the simulation button activates.	
S6.10	User presses the simulation button to verify the final result.	
S6.11	User uses the preview video player buttons to move around and see the result accordingly. For instance if the user presses the play button the sign language will be shown and deleted over the preview video in their respective times along with the video playback so as to simulate the final result.	
S6.12	If the sign language and the video are not in sync, user presses the offset button and enters the right entry time (TCin) that is required for the current subtitle. The same offset is applied to the rest of sign language times.	
S6.13	User presses the simulation button again to verify the final result.	
S6.14	User can also press the HMD simulation by pressing the corresponding button.	
S6.15	User puts on the HMD to watch the final result.	
S6.16	User can move up and down the sign language segment if specific ones need to be corrected.	
S6.17	The sign language is updated automatically in the Accessibility Content Manager database.	PUR.1.7.0

3.1.1.7 Object-Based Audio Editor

Table 11 - Scenario for object based audio editing

ID	Description	Requirements
S7.1	User opens the object-based audio editor.	
S7.2	User opens object-based audio scene or creates a new scene.	PUR.1.7.0 PUR.1.15.0
S7.3	User adds object-based audio objects to the scene (if necessary).	PUR.3.10.0
S7.4	User edits audio objects (if necessary).	
S7.5	User manually edits or adds tracks.	
S7.5.1	*User downloads AD track(s) from the Content Manager.*	
S7.5.2	*User imports AD track (Result from AD editor, with spatial metadata included).*	
S7.5.3	*User exports object-based audio scene.*	
S7.6	The spatial metadata is transmitted via a json file.	
S7.6.1	*User exports json configuration of AD objects from AD Editor.*	PUR.1.7.0
S7.6.2	*User exports raw audio files of AD tracks from AD Editor.*	
S7.6.3	*User exports OBA Scene file (ADM file) from OBA Editor without the AD.*	
S7.6.4	*User merges exports from AD Editor and OBA Scene into one scene with standalone software.*	

3.1.2 Accessibility Content Manager

3.1.2.1 Assignment of videos for the accessibility content production

Table 12 - Scenario for Assignment of videos for accessibility content production

ID	Description	Requirements
S8.1	User accesses the accessibility content manager web page via the web browser.	
S8.2	User enters username and password.	
S8.3	A window with the list of assets with its corresponding accessibility files appears.	
S8.4	User presses the new asset to upload the video to be subtitled, audio described or signed.	PUR.3.12.0
S8.5	User selects the new asset, edits metadata and presses the assign production button (subtitling, audio description or sign language).	PUR.3.1.0 PUR.3.2.0 PUR.3.3.0 PUR.3.14.0
S8.6	User assigns the corresponding accessibility content production to an operator (professional user that will have to produce the accessibility content).	PUR.3.13.0
S8.7	User repeats the process for the rest of videos.	

3.1.2.2 Getting an Accessibility Content file

Table 13 – Scenario for getting accessibility content files

ID	Description	Requirements
S9.1	User accesses the accessibility content manager web page via the web browser.	
S9.2	User enters username and password.	
S9.3	A window with the list of assets with its corresponding accessibility files appears.	
S9.4	User presses the find button and enters the criteria for the search.	
S9.5	A list of assets that comply with the criteria is displayed.	PUR.3.2.0
S9.6	User selects the asset and executes the download of the corresponding accessibility content file (subtitling, audio description or sign language).	PUR.3.1.0

3.1.3 Content Packager and Distribution

3.1.3.1 Generating Different audio formats with an audio renderer

Table 14 - Scenario for generating different audio formats

ID	Description	Requirements
S10.1	User loads object-based audio scene.	PUR.1.6.0 PUR.1.29.0
S10.2	User loads settings for rendering process and export.	
S10.3	User edits settings for rendering process and export (if necessary).	
S10.4	User defines export formats (first order ambisonics, binaural, stereo,...).	
S10.5	User starts rendering process.	PUR.1.14.0

3.1.3.2 Preparation of contents for the end user visualization

Table 15 – Scenario for Preparation of Contents for Distribution

ID	Description	Requirements
S11.1	Subtitling, audio description and sign language within the corresponding audio-visual contents are encoded and packaged.	PUR.3.4.0 PUR.3.5.0
S11.2	Signalling of the packaged contents to the client players.	PUR.3.6.0

3.1.3.3 Distribution

Table 16 – Scenario for Distribution

ID	Description	Requirements
S12.1	**VOD**: the content will be prepared as files on Content Delivery Network (CDN) storage ready to be distributed on a video on demand request.	PUR.3.7.0 PUR.3.8.0 PUR.3.9.0
S12.2	**VOD with TV linear distribution**: the content will be distributed to be consumed through companion screen synchronously to a linear TV content. In this case, a server may be required to package live or, as a more likely scenario, the content can be pre-packaged to be broadcast at a given time. Then the content is pushed to the CDN (may require authentication tokens which depend on the CDN of the broadcaster).	PUR.3.7.0 PUR.3.8.0 PUR.3.9.0

3.1.4 Player

In all the following use-cases the user is a home user and the requirements that each scenario satisfy are listed.

Table 17 - Scenario for playback

ID	Description	Requirements
S13.1	User accesses to a website in which a list of accessible 360º video contents (i.e., immersive contents enriched with accessibility assets) is available.	HUR.2.63.0
S13.2	User could use HMDs, smartphones and tablets for the playback of the immersive + accessible contents in both scenarios. The use of PCs (laptops, desktops…) is also possible.	HUR.2.6.0 HUR.2.38.0
S13.3	User presses the proper button/link to start the playback of the immersive + accessible contents. An interface provides control for the video playback.	HUR.2.54.0 HUR.2.62.0
S13.4	The player allows options for personalisation and stores the users preferences.	HUR.2.13.0 HUR.2.37.1 HUR.2.50.0 HUR.2.58.0
S13.5	User can enable the subtitle services, which allows adaptive and personalized presentation of the accessibility assets	HUR.2.8.0 HUR.2.24.0 HUR.2.25.0 HUR.2.27.0 HUR.2.28.0 HUR.2.29.0 HUR.2.30.0 HUR.2.31.1 HUR.2.31.2 HUR.2.50.1 HUR.2.51.0 HUR.3.4.0 HUR.3.5.0 HUR.3.11.0
S13.6	User can enable the audio description services, which allows an adaptive and personalized presentation of the accessibility assets	HUR.2.4.1 HUR.2.9.0 HUR.2.40.1 HUR.2.42.1 HUR.2.45.1 HUR.2.55.0 HUR.3.7.1
S13.7	User can enable the audio subtitle services, which allows an adaptive and personalized presentation of the accessibility assets	HUR.2.52.0 HUR.2.53.0 HUR.2.56.0
S13.8	User can enable the sign language service, which allows an adaptive and personalized presentation of the accessibility assets	HUR.2.18.0 HUR.2.19.0 HUR.2.21.0 HUR.2.21.2 HUR.2.49.0 HUR.2.49.1 HUR.2.60.0 HUR.3.1.0 HUR.3.9.0 HUR.3.12.0
S13.9	Users can dynamically activate/deactivate the presentation of the	HUR.2.16.0

	available accessibility assets, as well as to set the available personalisation options for each of them.	HUR.2.17.0 HUR.2.26.0 HUR.2.36.0 HUR.2.57.0
S13.10	During playback, user can freely explore the 360º area (e.g., by moving the head, using the touch screen, mouse…), and the presentation of the accessibility assets will be adapted accordingly.	HUR.2.61.0 HUR.3.2.0 HUR.3.6.0
S13.11	The player can be controlled by voice commands and provides spoken feedback	HUR.2.10.0 HUR.2.10.1 HUR.2.59.0

3.2. Logical Views

This section provides a description of the logical view of the architecture. We describe the most important classes, their organization in service packages and subsystems, and the organization of these subsystems into layers. It also describes the most important use-case realizations, for example, the dynamic aspects of the architecture. Class diagrams may be included to illustrate the relationships between architecturally significant classes, subsystems, packages and layers.

The logical view of the ImAc system is comprised of the main packages: Editor Tools (WP4) Content Manager (T3.2), Content Packager / Distributor (T3.3) and Player (T3.5)

3.2.1 System overview

Figure 3 shows the logical architecture that developed in the platform specification D2.3. It shows each of the different modules that interact with each other to execute the necessary processes for the creation, management and distribution of audiovisual content and accessible services.

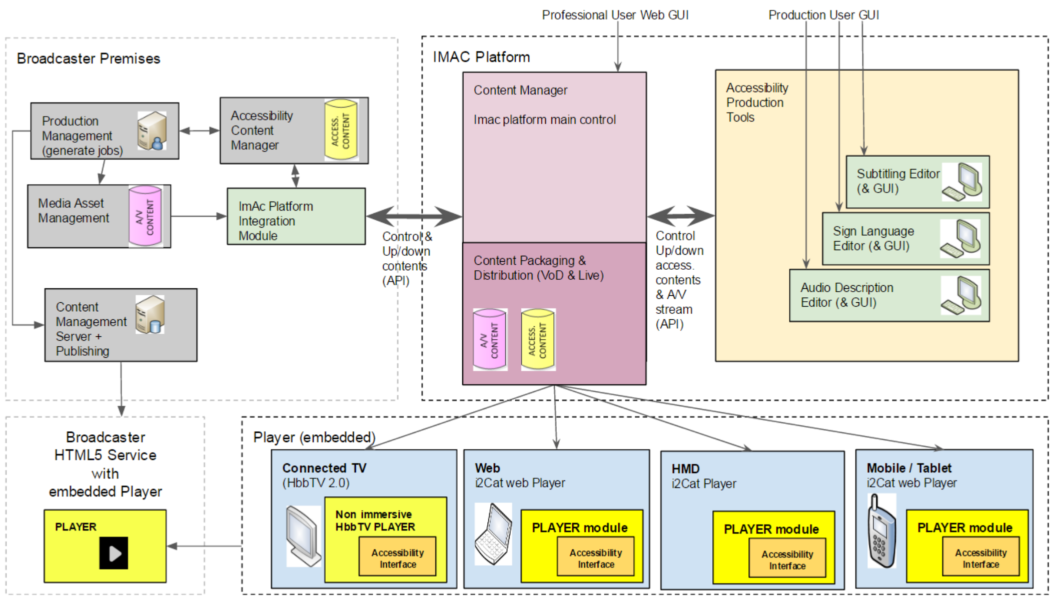

Professional User Web GUI
Production User GUI
Broadcaster Premises
Production Management (generate jobs)
Accessibility Content Manager
ACCESS. CONTENT
Media Asset Management
A/V CONTENT
ImAc Platform Integration Module
Content Management Server + Publishing
Control & Up/down contents (API)
IMAC Platform
Content Manager
Imac platform main control
Content Packaging & Distribution (VoD & Live)
Control Up/down access contents & A/V stream (API)
Accessibility Production Tools
Subtitling Editor (& GUI)
Sign Language Editor (& GUI)
Audio Description Editor (& GUI)
Broadcaster HTML5 Service with embedded Player
PLAYER
Player (embedded)
Connected TV (HbbTV 2.0)
Non immersive HbbTV PLAYER
Accessibility Interface
Web i2Cat web Player
PLAYER module
HMD i2Cat Player
Mobile / Tablet i2Cat web Player

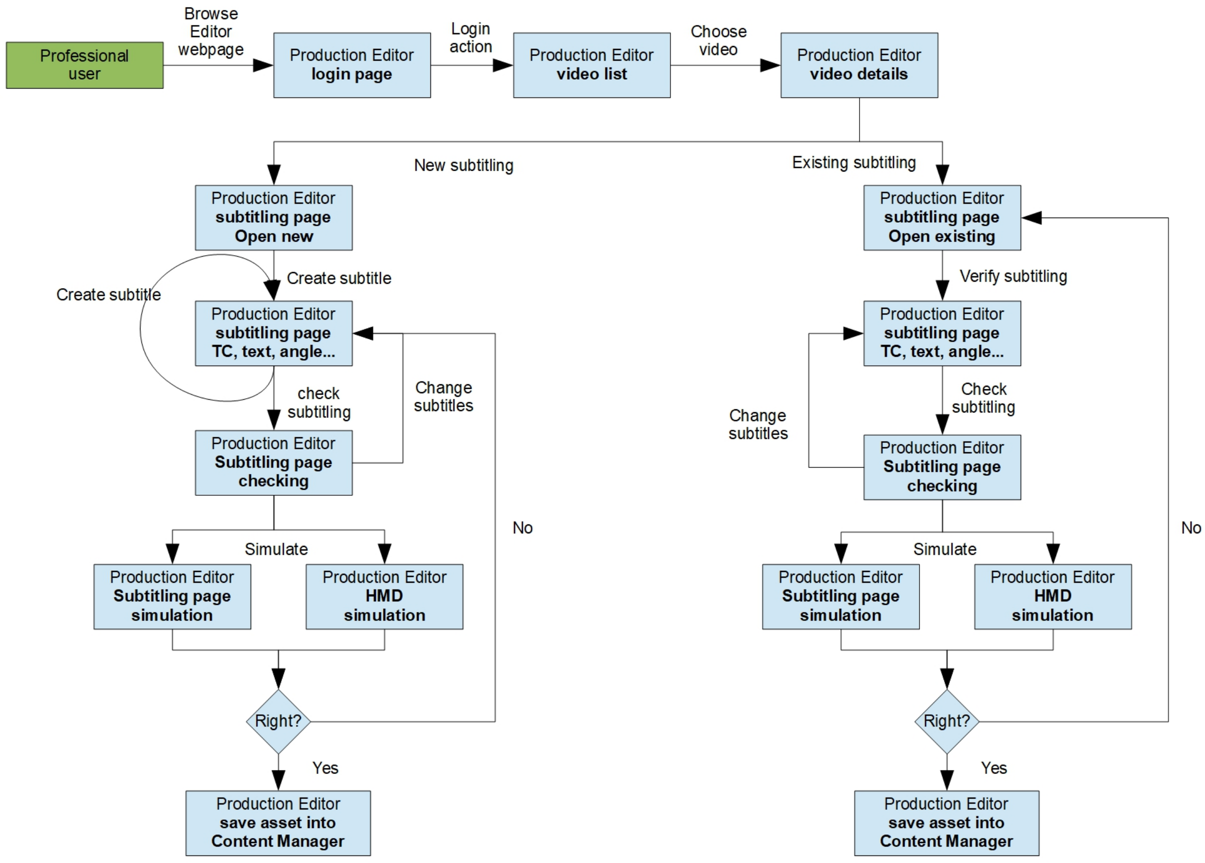
Figure 3 - The logical architecture of the ImAc project

The architecture is divided in 3 main areas:

- Broadcast Premises - an OTT platform owner or any content owner with the requirement to produce accessibility content for its audio-visual immersive content could also represent this area. A general overview of internal broadcaster workflows is shown, with the following modules:
    - Production Management module from where the different works of accessible content production are requested.
    - Accessibility Content Manager module to manage and store the ImAc produced contents.
    - Media Asset Management platform manages broadcaster media workflows, systems and assets throughout the multimedia production & distribution chain.
    - Content Management System + Publishing module allows the management, packaging and publishing to Internet of broadcast contents.
    - ImAc Platform Integration Module will be in charge to interconnect broadcaster workflows with ImAc platform and will allow all the necessary communication to execute all required processes through a pre-defined API (Application Programming Interface).
- Player (embedded) - The ImAc media player will be developed within the ImAc project ready to play all immersive accessible services and with an accessibility interface to ease the access to end users. This ImAc media player will be embedded in the web service of the broadcaster or OTT platform owner.
- ImAc Platform - this area brings together all the main modules for the management, production, packaging and distribution of accessible contents and will be detailed in next sections.

3.2.1 Production Editors

Figure 4 provides a logical view diagram for the subtitling editor, Figure 5 provides a logical view diagram for the audio description editor and Figure 6 provides a logical view diagram for the sign language editor.



Professional user
Browse Editor webpage
Production Editor login page
Login action
Production Editor video list
Choose video
Production Editor video details
New subtitling
Existing subtitling
Production Editor subtitling page Open new
Create subtitle
Create subtitle
Production Editor subtitling page TC, text, angle...
check subtitling
Change subtitles
Production Editor Subtitling page checking
Simulate
Production Editor Subtitling page simulation
Production Editor HMD simulation
No
Right?
Yes
Production Editor save asset into Content Manager
Production Editor subtitling page Open existing
Verify subtitling
Production Editor subtitling page TC, text, angle...
Check subtitling
Change subtitles
Production Editor Subtitling page checking
Simulate
Production Editor Subtitling page simulation
Production Editor HMD simulation
No
Right?
Yes
Production Editor save asset into Content Manager

Figure 4 - Logical View Diagram - Subtitling Editor

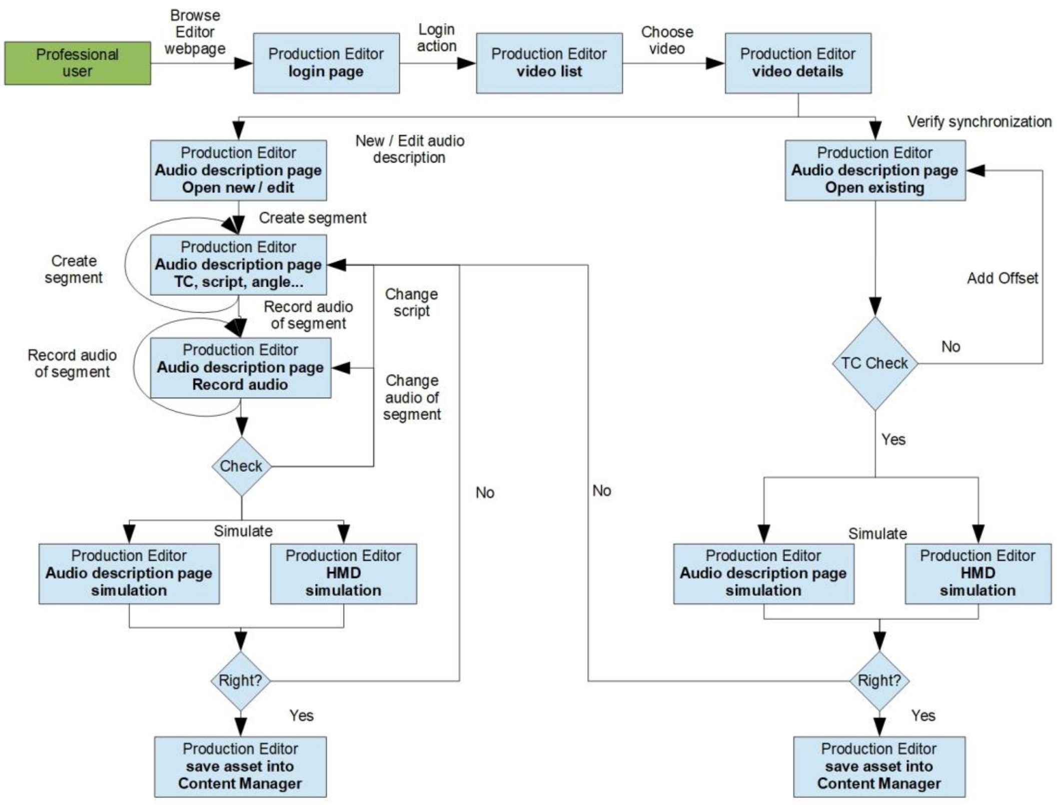

Professional user
Browse Editor webpage
Production Editor login page
Login action
Production Editor video list
Choose video
Production Editor video details
New / Edit audio description
Verify synchronization
Production Editor Audio description page Open new / edit
Production Editor Audio description page Open existing
Create segment
Production Editor Audio description page TC, script, angle...
Record audio of segment
Production Editor Audio description page Record audio
Change script
Change audio of segment
Add Offset
TC Check
No
Yes
Check
Simulate
Production Editor Audio description page simulation
Production Editor HMD simulation
Right?
Yes
Production Editor save asset into Content Manager

Figure 5 - Logical View Diagram - Audio Description Editor

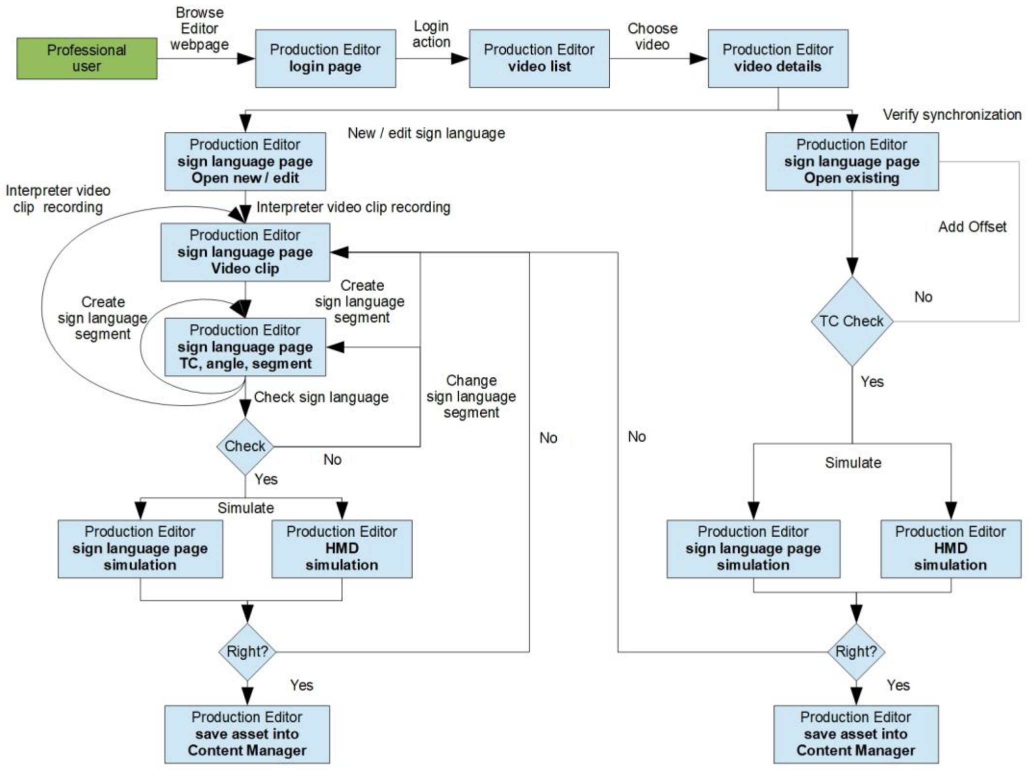

Professional user
Browse Editor webpage
Production Editor login page
Login action
Production Editor video list
Choose video
Production Editor video details
New / edit sign language
Verify synchronization
Production Editor sign language page Open new / edit
Production Editor sign language page Open existing
Interpreter video clip recording
Production Editor sign language page Video clip
Create sign language segment
Production Editor sign language page TC, angle, segment
Check sign language
Change sign language segment
Add Offset
TC Check
Check
No
Yes
Simulate
Production Editor sign language page simulation
Production Editor HMD simulation
Right?
Production Editor save asset into Content Manager

Figure 6 - Logical View Diagram Sign Language Editor

3.2.2 Accessibility Content Manager

Figure 7 provides a logical view diagram for the accessibility content manager.

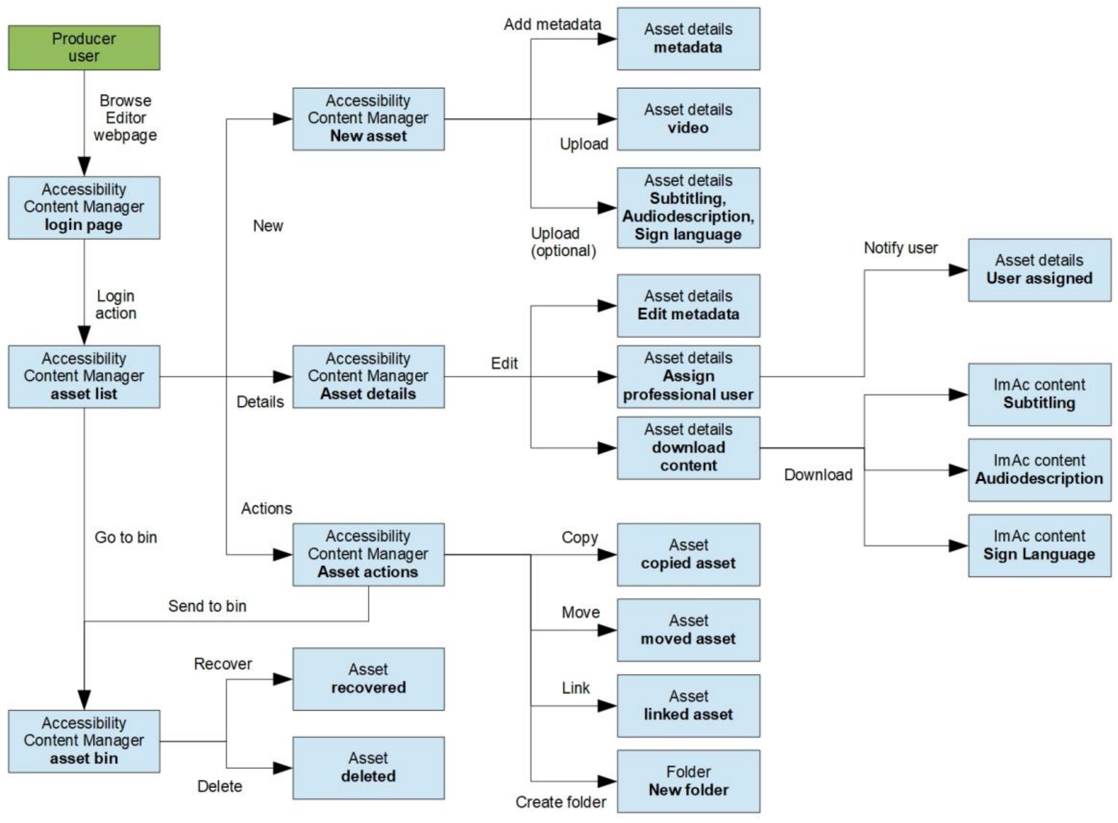

Producer user
Browse Editor webpage
Accessibility Content Manager login page
Login action
Accessibility Content Manager asset list
New
Details
Actions
Go to bin
Send to bin
Accessibility Content Manager New asset
Accessibility Content Manager Asset details
Accessibility Content Manager Asset actions
Add metadata
Upload
Upload (optional)
Asset details metadata
Asset details video
Asset details Subtitling, Audiodescription, Sign language
Edit
Asset details Edit metadata
Asset details Assign professional user
Asset details download content
Notify user
Asset details User assigned
Download
ImAc content Subtitling
ImAc content Audiodescription
ImAc content Sign Language
Copy
Move
Link
Create folder
Asset copied asset
Asset moved asset
Asset linked asset
Folder New folder
Recover
Delete
Accessibility Content Manager asset bin
Asset recovered
Asset deleted

Figure 7 - Logical View Diagram - Accessibility Content Manager

3.2.3 Content Packager and Distribution

Figure 8 shows a logical View of the content packager and distribution module. This module will communicate with the Content Manager in a close way since the Content Manager will pilot it. The ImAc project will define protocols to ensure communications between the modules following the best practices of the industry.

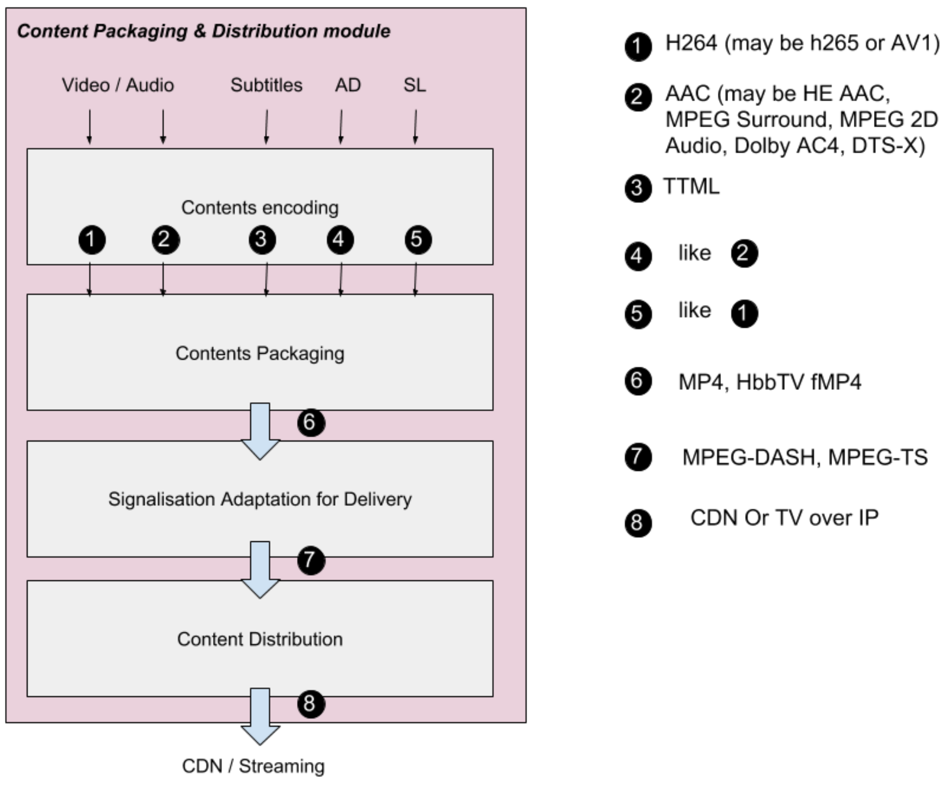


Figure 8 - Logical View Diagram - Content packager and distribution module

This module will match four main required functionalities:

- Firstly, it will encode the content (compress the audio and video).
  - Video: h.264. Maybe h.265 or AV1 depending on technical and standardization advances.
  - Audio (including AD): AAC. Maybe other codecs or extensions (HE-AAC/MPEG Surround, MPEG 2D Audio, Dolby AC4, DTS-X) depending on SDK availability since most of these codecs are proprietary or heavily patented.
  - Subtitles: TTML (v1, IMSC1 profiles) (or WebVTT if standardization occurs).
  - SL: (MP4)
- Secondly, it will package the content inside a suitable container for delivery, and it will ensure that the content is segmented appropriately.
  - TV over IP: MP4
  - TV: HbbTV.

    Note: we may want to use the CMAF ISOBMFF application format.
- Thirdly, this module will ensure a proper delivery that includes the signalling of the metadata according the editors' wishes.
  - MPEG-DASH (DASH-IF and HbbTV industry profiles)
  - MPEG-TS (according to HbbTV)
- Finally, it will push the content to the appropriate network (whether using streaming protocols, or file-based caching infrastructures).
  - CDN only.
  - TV over IP for TV.

Figure 9 demonstrates the delivery mechanism for the ImAc content. Two video streams are transmitted which include the main video content and the sign language, both using MPEG-DASH. In addition to this the subtitle data is transmitted using IMSC TTML and the audio is delivered in each of the formats available for the content.

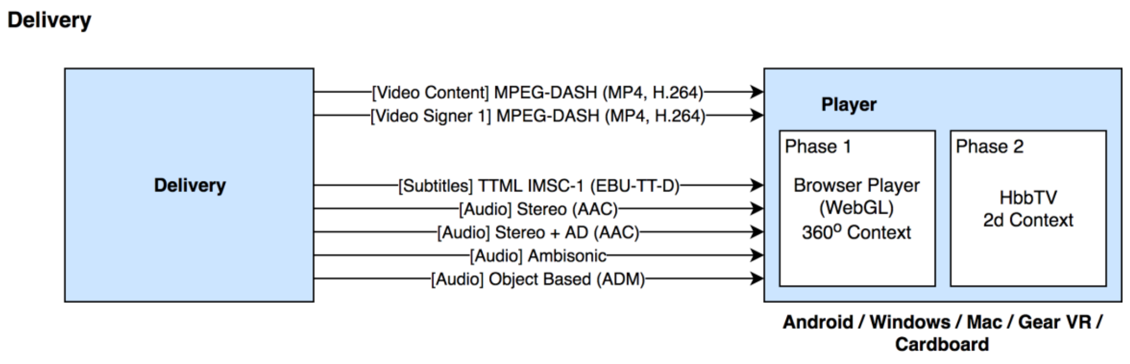

Delivery
Delivery
[Video Content] MPEG-DASH (MP4, H.264)
[Video Signer 1] MPEG-DASH (MP4, H.264)
[Subtitles] TTML IMSC-1 (EBU-TT-D)
[Audio] Stereo (AAC)
[Audio] Stereo + AD (AAC)
[Audio] Ambisonic
[Audio] Object Based (ADM)
Player
Phase 1
Browser Player (WebGL) 360º Context
Phase 2
HbbTV 2d Context
Android / Windows / Mac / Gear VR / Cardboard

Figure 9 - Deployment View Diagram – Delivery

When possible all content will be provided with OBA. This is an ideal situation as the both ambisonics and a basic stereo mix can be down mixed from OBA, meaning that a single audio source can be used to generate each of the different formats that could be required by different players – as shown in Figure 10.

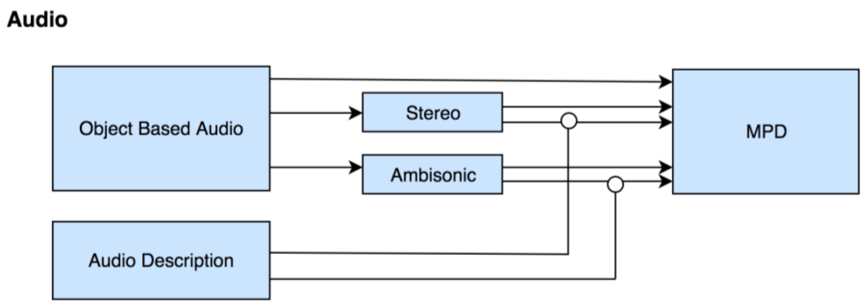


Figure 10 - Deployment View Diagram – Audio

Figure 11 shows a high level overview of how the content is transmitted from the content manager through to the CDN server for distribution.

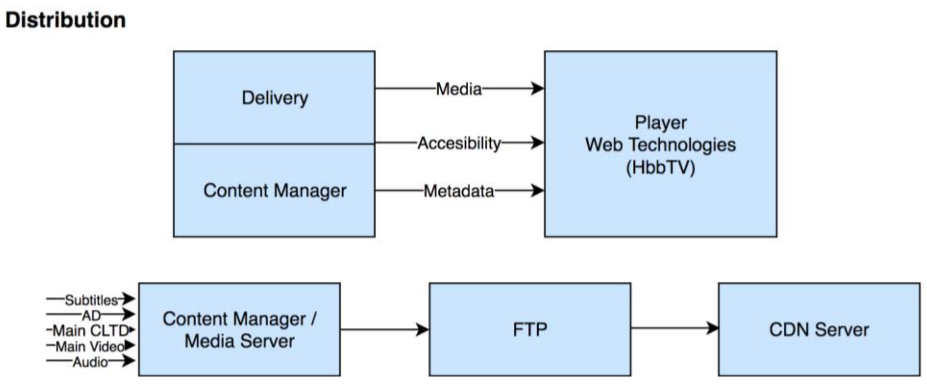


Figure 11 - Deployment View Diagram - Distribution

3.2.4 Player

Figure 12 shows a high-level overview of the logical view for the player. It illustrates the interactions between the user and the player, as well as of how the presentation of the immersive and accessibility contents can be dynamically activated/deactivated and adapted according to tracking functions (e.g. users' movements, interactions…) and to the setting of the available personalisation features.

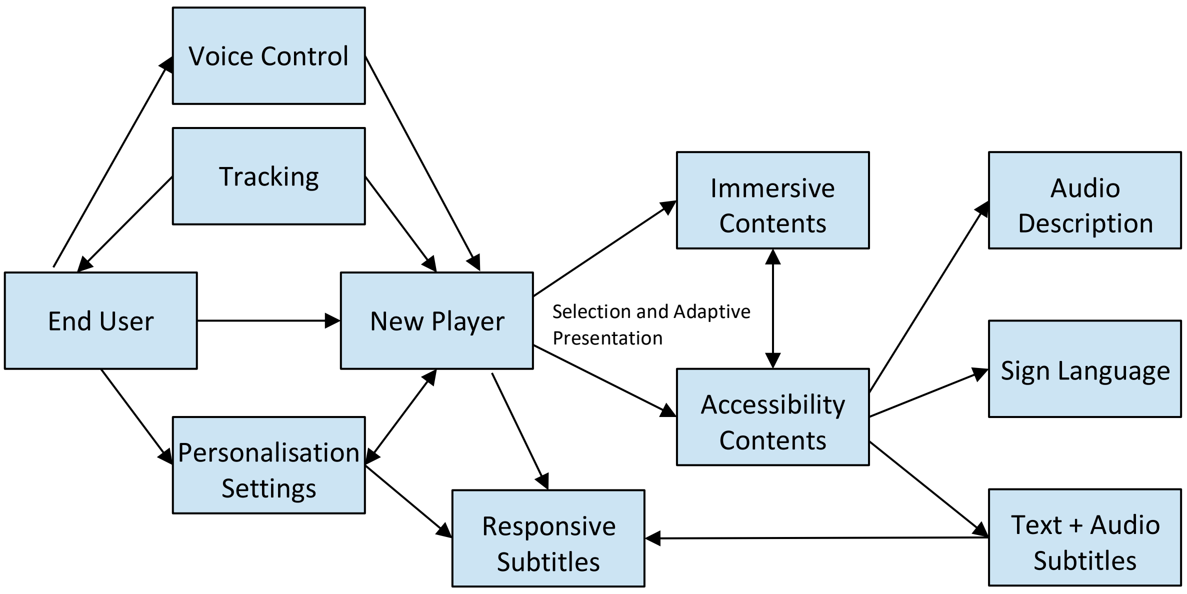


Figure 12 - Logical View for the Player

In relation to this, Figure 13 illustrates the main sub-systems, layers, modules and libraries that make up the ImAc player, together with the relationships and interactions between these components. The names of the involved JS libraries implementing the functionalities of these components are also indicated. By using these libraries, the instances of the classes implementing each of the ImAc player's requirements are created. In the Figure, AV stands for Audio+Video, V for Video, A for Audio, SL for Sign Language, S for Subtitles, and DVB-CSS for Digital Video Broadcasting - Companion Screens and Streams.

More details can be found in D3.5.

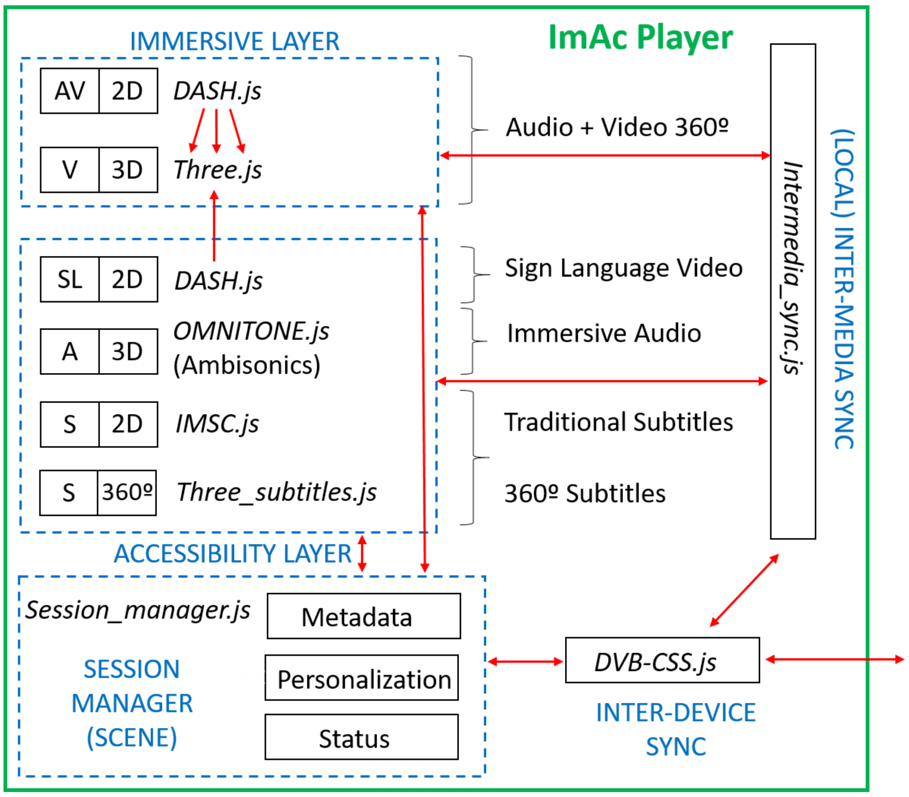
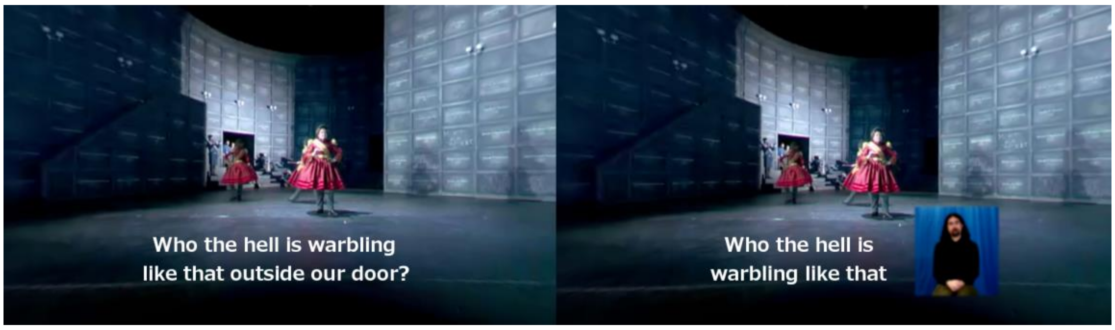


Figure 13 - Sub-systems, layers and modules composing the player

3.2.5 Responsive Subtitles

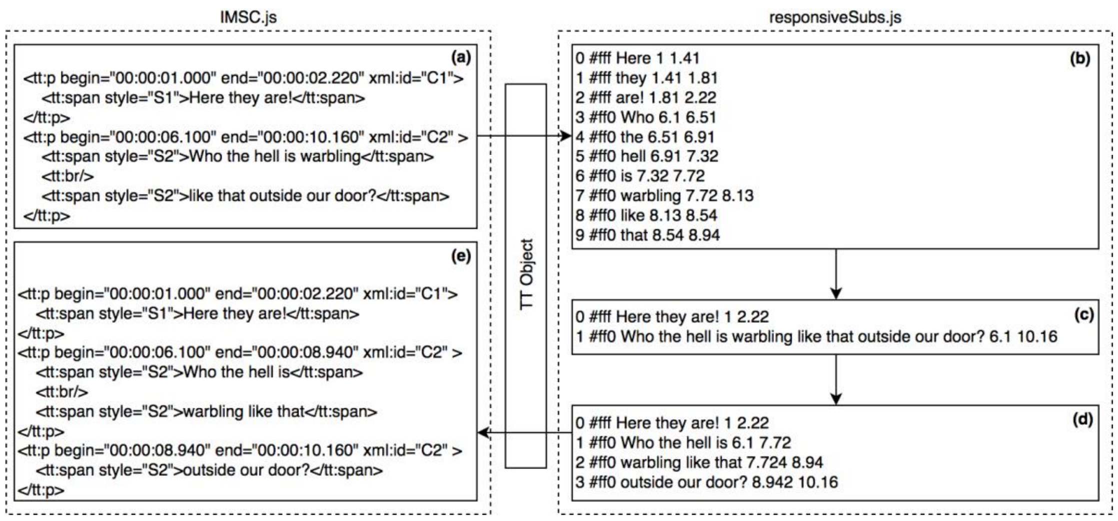
The subtitle presentation methods are limited by preserving the structure of the subtitles file. Using a responsive subtitle approach [2] allows further customisation by providing rules, which allow the subtitles to be dynamically re-structured. This approach is particularly effective when adapting content from traditional television displays into an immersive environment, such as rendering the subtitle as speech bubbles attached to a character, or for instance if you if you wish to reduce the width of the subtitles in order to make room for graphics as shown in Figure 14.



Who the hell is warbling like that outside our door?
Who the hell is warbling like that

Figure 14: Responsive subtitles allow the rendering area to be dynamically changed to make space for graphics, or other accessibility services such as a signer.

As part of the ImAc project we have followed practices used in responsive web design to prototype a JavaScript library for generating responsive subtitles. This adopts the principles of text flow and line length informed by semantic mark-up along with styles to control the final rendering. The library provides an extension to IMSC.js (a JavaScript library for rendering IMSC1 Text and Image Profile documents to HTML5) where IMSC documents can be loaded into a TT-object. Our responsive subtitle library restructures a TT-object based on line character width and line count as shown in Figure 15.



```
<tt:p begin="00:00:01.000" end="00:00:02.220" xml:id="C1">
   <tt:span style="S1">Here they are!</tt:span>
</tt:p>
<tt:p begin="00:00:06.100" end="00:00:10.160" xml:id="C2" >
   <tt:span style="S2">Who the hell is warbling</tt:span>
   <tt:br/>
   <tt:span style="S2">like that outside our door?</tt:span>
</tt:p>
```

```
0 #fff Here 1 1.41
1 #fff they 1.41 1.81
2 #fff are! 1.81 2.22
3 #ff0 Who 6.1 6.51
4 #ff0 the 6.51 6.91
5 #ff0 hell 6.91 7.32
6 #ff0 is 7.32 7.72
7 #ff0 warbling 7.72 8.13
8 #ff0 like 8.13 8.54
9 #ff0 that 8.54 8.94
```

```
0 #fff Here they are! 1 2.22
1 #ff0 Who the hell is warbling like that outside our door? 6.1 10.16
```

```
0 #fff Here they are! 1 2.22
1 #ff0 Who the hell is 6.1 7.72
2 #ff0 warbling like that 7.724 8.94
3 #ff0 outside our door? 8.942 10.16
```

```
<tt:p begin="00:00:01.000" end="00:00:02.220" xml:id="C1">
   <tt:span style="S1">Here they are!</tt:span>
</tt:p>
<tt:p begin="00:00:06.100" end="00:00:08.940" xml:id="C2" >
   <tt:span style="S2">Who the hell is</tt:span>
   <tt:br/>
   <tt:span style="S2">warbling like that</tt:span>
</tt:p>
<tt:p begin="00:00:08.940" end="00:00:10.160" xml:id="C2" >
   <tt:span style="S2">outside our door?</tt:span>
</tt:p>
```

Figure 15 (a) IMSC.js converts the TTML document into a TT Object. (b) The responsive library atomizes the TT object into words, preserving an interpolated time and style for each word. (c) The words are reconstructed into phrases split by a pause in the dialogue or a change of speaker. (d) The phrases are subdivided using a best-fit algorithm to meet the line length requirements. (e) a new TT objet is generated with IMSC.js.

By working directly with TT-objects allows this library to be simply connected to any application which already uses the standard IMSC.js implementation allowing customization controls to be retrofitted, such as font size as shown in Figure 16. Subtitles are re-blocked by adhering to the number of characters that can fit into the display container at the chosen font size. Firstly each paragraph is recombined, based on a unique speaker. A best-fit algorithm then breaks each paragraph up to individual captions in order to fit the container. Due to the nature of the changing font size this may provide more or less captions than the original subtitles, however as the number of words is remains the same the reading speed never changes. As words are evenly distributed it also avoids leaving orphaned words.

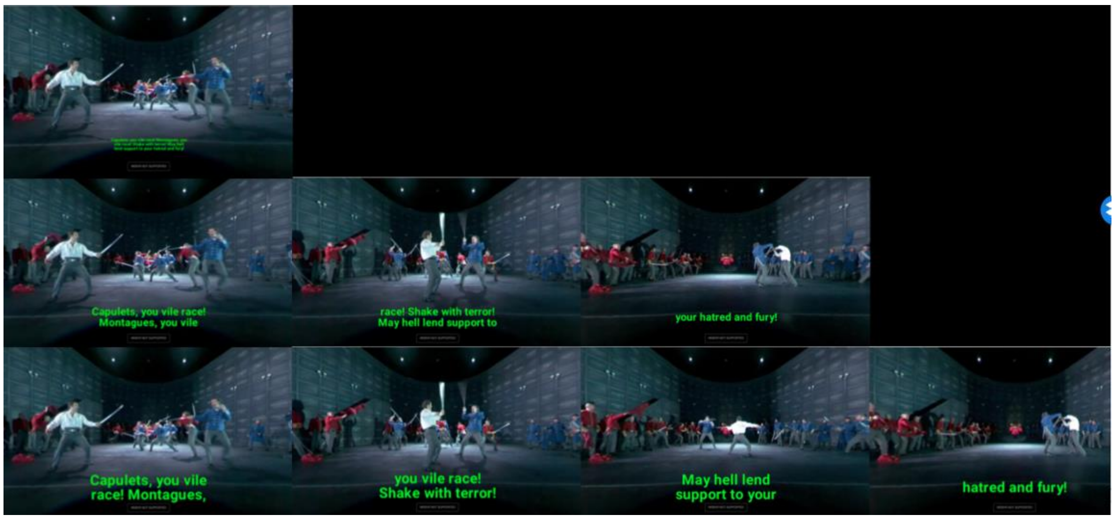


Figure 16: Responsive subtitles re-blocked based on consumers font size requirements. The subtitles are presented differently when the user chooses a font size of 50%, 100%, 125% and 150%.

3.2.6 Voice Control

As defined in requirement HUR.02.11.0 out the ImAc player must support voice control and therefore the platform will integrate with existing voice control hardware (such as the Amazon Echo) for Voice Control.

Table 18– The functionality list derived from the interface design that needs to be integrated with the voice commands

Command	Action	State
Play / Pause	Play/Pause Toggle	With content queued up / playing
Skip back	Skip Backward	With content playing
Skip forward	Skip Forward	With content playing
Volume Up	Increases volume	In any mode. NOTE: With submenu of AD and AST open, this command controls the volume of the respective service
Volume Down	Decrease volume	In any mode. NOTE: With submenu of AD and AST open, this command controls the volume of the respective service
Subtitle	Open ST submenu	Playing content which has subtitles
Signer	Open signer submenu	Playing content which has signer
Audio description	Open AD submenu	Playing content which has AD
Audio subtitling	Open AST submenu	Playing content which has AST
On/Off	Switch on/off service	With submenu of one of the services open (ST, SL, AD, AST)
Easy to read on/off	Switch on/of easy-to-read mode of subtitles	With submenu of ST or AST open
Language	Open language submenu	With any submenu open that has language as a customization category (ST, AD, AST, general settings)

English / Deutsch / Català/ Español	Select language of service or of interface	With any of the language submenus open (ST, AD, AST, general settings) NOTE: Easy-to-read option is only available for ST and AST
Position	Open position submenu	With any of the position submenus open (ST, SL)
Top / bottom	Select position of subtitles	With subtitle position submenu open
Left / right	Select position of signer	With signer position submenu open
Background	Open background submenu	With subtitle submenu open
Semi-transparent / Outline	Select background of subtitles	With subtitle background submenu open
Size	Open subtitle size submenu	With subtitle submenu open
Small / medium / large	Select size of subtitles	With subtitle size submenu open
Speaker localization	Open guiding mechanism submenu	With any submenu open that has guiding mechanism as a customization category (ST, SL)
None / arrow / radar / auto	Select guiding mechanism	With any of the guiding mechanism submenus open (ST, SL). NOTE: radar option is only available for ST, forced perspective option is only available for SL
Area of display	Open comfort field of view submenu	With any submenu open that has comfort field of view as a customization category (ST, SL)
Small / medium / large	Select comfort field of view	With any of the comfort field of view submenus open (ST, SL). NOTE: The actual sizes behind the commands "small/medium/large" are not the same for the two services
Presentation mode	Open presentation mode submenu	With AD submenu open
Action / Head position / Soundscape	Select presentation mode for AD	With AD presentation mode submenu open
Settings	Open settings menu	In any mode
User profile	Open user profile submenu	With settings menu open
Save settings	Save currents settings of interface as user profile	With user profile submenu open
Load profile	Load user profile	With user profile submenu open
Next	Go to next menu group	With circular menu open (low-sighted mode)
Exit	Exit menu	With any menu open

Table 19 – Examples of the command words presented in English, Catalan, Spanish and German and the corresponding JS function.

English	**Catalan**	**Spanish**	**German**	***JS command / function name***
Volume Up	Apujar Volum	Subir Volumen	Lauter	getChangeVolumeFunc(true)
Volume Down	Abaixar Volum	Bajar Volumen	Leiser	getChangeVolumeFunc(false)
Play			Wiedergabe / Fortsetzen	getPlayPauseFunc(true)
Pause	Pausar	Pausar	Pause	getPlayPauseFunc(false)
Seek Forward	Avançar	Avanzar	Weiter	getSeekFunc(true)
Seek Back	Retrocedir	Retroceder	Zurück	getSeekFunc(false)
Subtitles On	Activar subtítols	Activar subtítulos	Untertitel an	getOnOffFunc(‘subtitlesOffButton’)
Subtitles Off	Desactivar subtítols	Desactivar subtítulos	Untertitel aus	getOnOffFunc(‘subtitlesOnButton’)
Open Menu	Obrir Menú	Abrir Menu	Öffne Menü	getOpenMenuFunc()
Close Menu	Tancar Menú	Cerrar Menu	Schließe Menü	getCloseTradMenuFunc()

The workflow of the voice control component is shown in Figure 17 and described below:

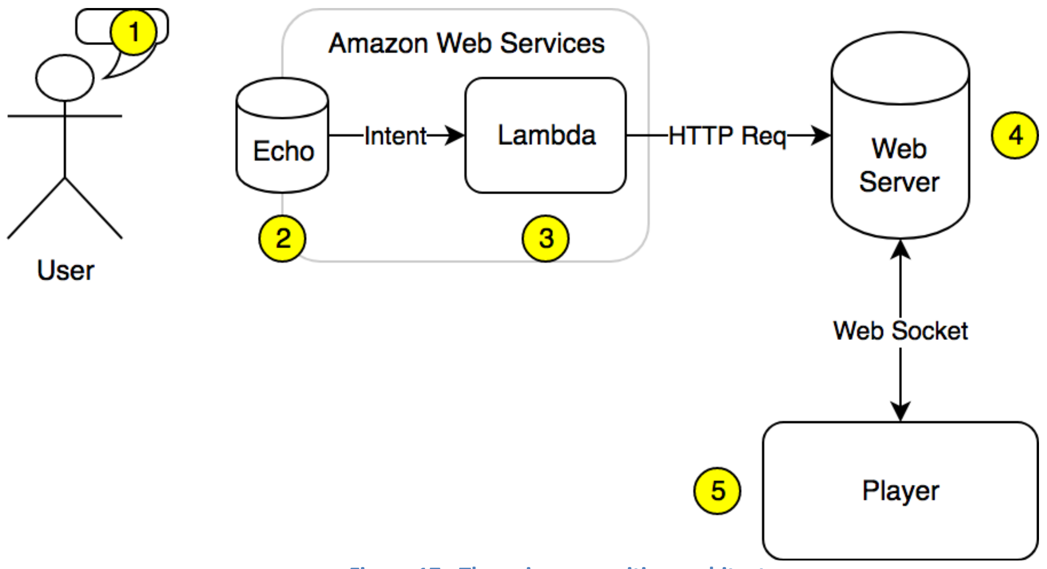


Figure 17 - The voice recognition architecture

1. The User

The user issues commands to Alexa. An application built for Alexa is referred to as a 'skill' and you begin interacting with a skill by issuing the command '[Wake Word] open [skill invocation name]. In the prototype the wake word was set to 'echo' and the skill invocation name was set to 'ImAc' so in this demo the user would start the interaction by issuing the command 'Echo open ImAc'.

2. The Echo

The Alexa skills are built within the Amazon Web services framework (developer.amazon.com). Each skill contains a number of 'Intents' where each 'intent' is designed to trigger a specific event, however there maybe multiple phrases that could be used to derive the same action. Variables such as numbers can also be defined within intent. The 'intents' are defined using a JSON document.

The intents are each defined with a unique name, and a set of sample phrases which could be used to trigger them:

```
{
        "name": "playVideoIntent",
        "slots": [],
        "samples": [
            "play"
        ]
    },
```

When capturing a user defined value, such as a specific subtitle size, the slots can be defined as a wildcard for where the values would go:

```
 "name": "subsSizeIntent",
"slots": [
    {
        "name": "subSize",
        "type": "AMAZON.NUMBER"
    }
],
"samples": [
    "change subtitle size to {subSize}",
    "set subtitle size {subSize}",
    "subtitle size {subSize}"
]
```

3. AWS Lambda

The identified intend name is posted to Lambda. This is an AWS application, which allows you to run JavaScript code within AWS in a server-less manner. In essence what this does is to bridge each intent from the Echo and forward it directly to our webserver. It also formulated a response, which is the spoken response returned to the User. It does this by sending an HTTP POST request to the webserver with the current intent.

4. Web Server

The webserver receives the request and forwards it to an open web socket in the player. There will also need to be a device registration step.

5. Player

node.js and web sockets are used to allow the server to send commands directly to the listening client.

3.3. Process Views

This section provides a description of the process view of the architecture. It describes the tasks (processes and threads) involved in the system's execution, their interactions and configurations. It also describes the allocation of objects and classes to tasks.

3.3.1 Production Editors processes

3.3.1.1 Subtitling editor processes

- Video player process: render the received 360 video from the server for the preview player.
- Subtitling simulation process:
    - Each time a subtitle is edited it is previewed on top of the video.

    - When the subtitling is finished a simulation of the complete subtitling is done on top of the video while playing back the video.
- HMD simulation process: when the subtitling is finished a simulation of the complete subtitling is done on the HMD.

3.3.1.2 Audio description editor processes

- Video player process: render the received 360 video from the server for the preview player.
- Audio description simulation process:
    - Each time an audio description segment is edited a mixed audio preview can be played back for the verification.
    - When the audio description is finished a simulation of the complete audio description is done along the playback of the video.
- HMD simulation process: when the audio description is finished a simulation of the complete audio description is done on the HMD.

3.3.1.3 Sign language editor processes

- Video player process: render the received 360 video from the server for the preview player.
- Sign language simulation process:
    - Each time a sign language segment is edited a preview of the corresponding video clip can be played back for the verification.
    - When the sign language is finished a preview of the sign language in a separate window is done along the playback of the main video.
- HMD simulation process: when the sing language is finished a simulation of the complete sign language is done on the HMD.

3.3.1.4 Processes tasks of object-based audio editor

- Object-based audio editor task: Create or edit object based audio scenes. This includes import of various audio formats (e.g. stereo, 5.1, Ambisonics) to convert them into object based audio scene format.
- Content uploader task (push content to audio renderer): Export final audio scene and transfer it to the audio renderer.

3.3.2 Accessibility Content Manager processes

Accessibility Content Manager processes:

- Video uploading process: when a video is uploaded it is stored in the database with the respective metadata.
- Accessibility content uploading process: when accessibility content file is uploaded it is associated with the corresponding video in the database.
- Accessibility content edition process: when a accessibility content file is uploaded from the editors, it is updated in the database.

- Accessibility content delivery: when accessibility content file is requested from the web service, it is converted into the corresponding file format and delivered.

3.3.3 Content Packager and Distribution

To have a better understanding of the processes for contents creation and preparation, the schema in Figure 18 indicates the different sequential steps, and thus the content workflow, through the different modules of the platform. These steps are also listed below

1. AV content is ingested through the SFTP
2. AV content is transcoded for the ACM
3. Accessibility Content is authored with the ACM (ST, AD,SL)
4. Accessibility & AV content is packaged
5. Packaged content is then converted into DASH format (i.e. encoded in multi-qualities, segmented) and the appropriate metadata files are created (i.e., MPD, and updated list of available contents) and published on the webserver for the players

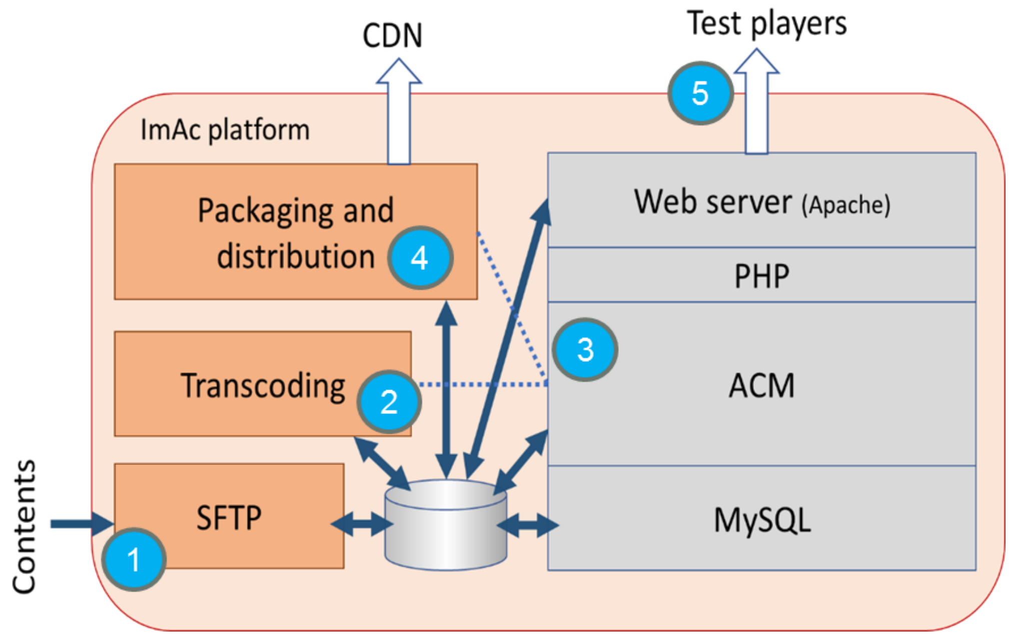


Figure 18: Content workflow through the platform

In order to detail more the workflow and the underlying process, a sequence diagram is presented below in Figure 19. This diagram allows following the content transformation through the different processes running onto the platform and the events sent between processes or external actors (Concepts and implementation details are better explained in section 3).

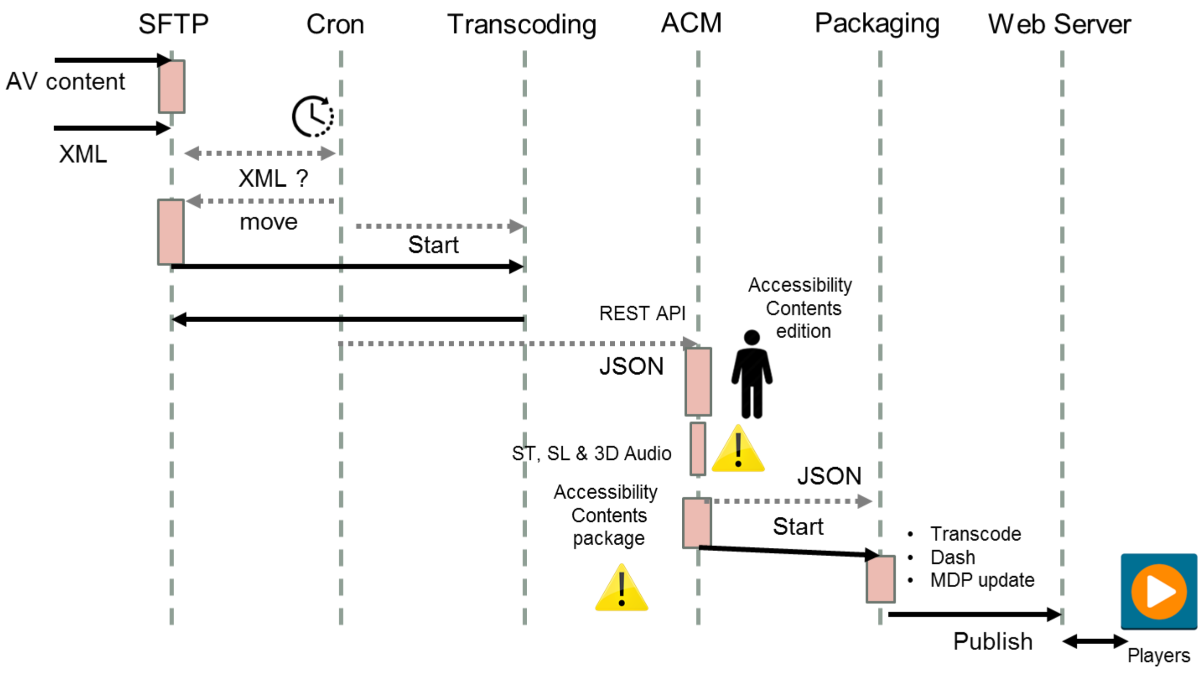


Figure 19: Sequence diagram of the content processing

In this diagram, a step is not described during publishing phase: Updating of the available contents (JSON file) when DASHing new contents.

In the final platform, all the content is supposed to be processed automatically, except for the ACM phase, where the accessibility contents have to be (co-)authored.

3.3.4 Player processes

End-users create an instance of the media player by opening a web browser, typing the target URL and selecting the appropriate contents or by directly associating a companion device with a main TV (in S13.1). The selection or direct playback of immersive contents will enable the Immersive Layer of the player, which includes the functionalities for creating 360º scenes, by using *three.js* library, from instances of traditional 2D media players, by using the *DASH.js* library. The Accessibility Layer will be enabled through controls of the User Interface, although it may be also automatically enabled based on the user's profile. That layer includes modules for the presentation of the considered accessibility assets, namely: textual and audio subtitles, sign language videos, and audio description.

When more than one media component is being simultaneously presented, their spatial and temporal relationships must be preserved during playback. A new developed *intermedia_sync.js* library needs to be developed to achieve this goal.

Moreover, key functionalities of the player will be provided by the Session Manager module, implemented in the session_manager.js library. That module will store relevant information about the session, such as the metadata obtained from the Content Manager (e.g., describing the available contents, their relationships, etc.), the contents being currently presented, the available personalisation options together with the current settings, as well as the status of the session (e.g., elapsed time, duration, other active devices...).

3.4. Deployment Views (Physical Views)

A description of the deployment view of the architecture describes the various physical nodes for the most typical platform configurations. This section also describes the allocation of tasks (from the Process View) to the physical nodes.

This section is organized by physical network configuration; a deployment diagram, followed by a mapping of processes to each processor, illustrates each such configuration.

3.4.1 Production Editors

The production editors require a Desktop PC for object-based audio editor.

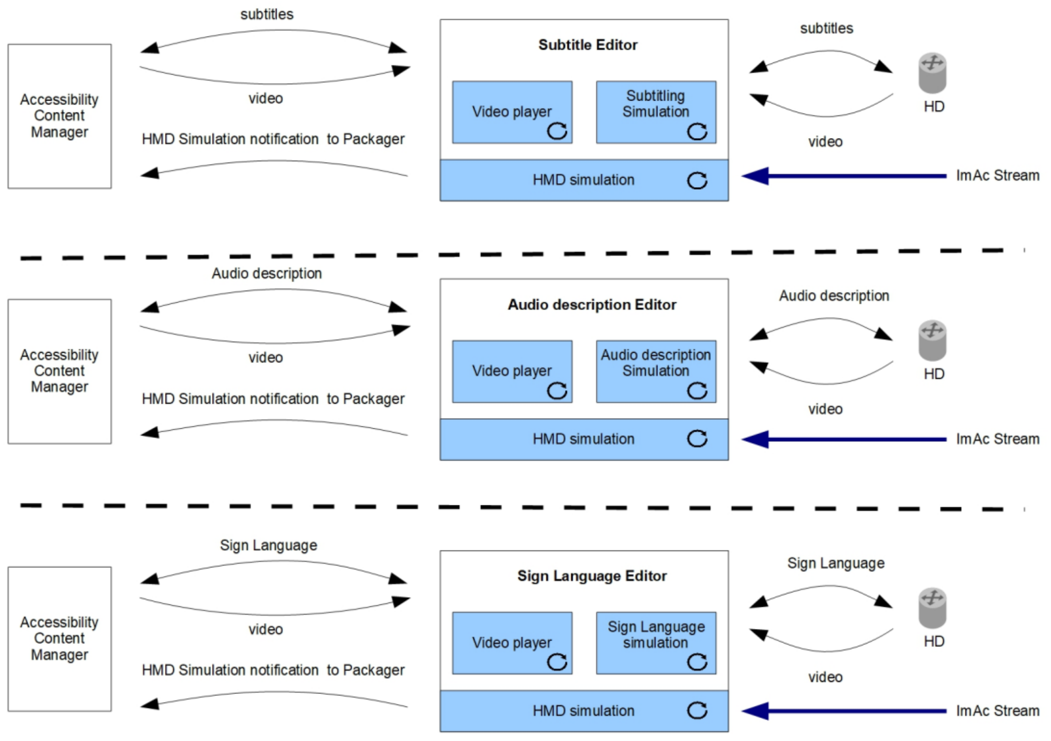
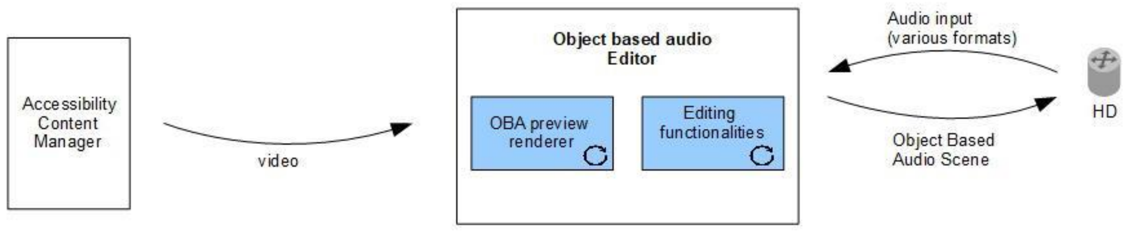


Figure 20 - Deployment View Diagram - Production Editors

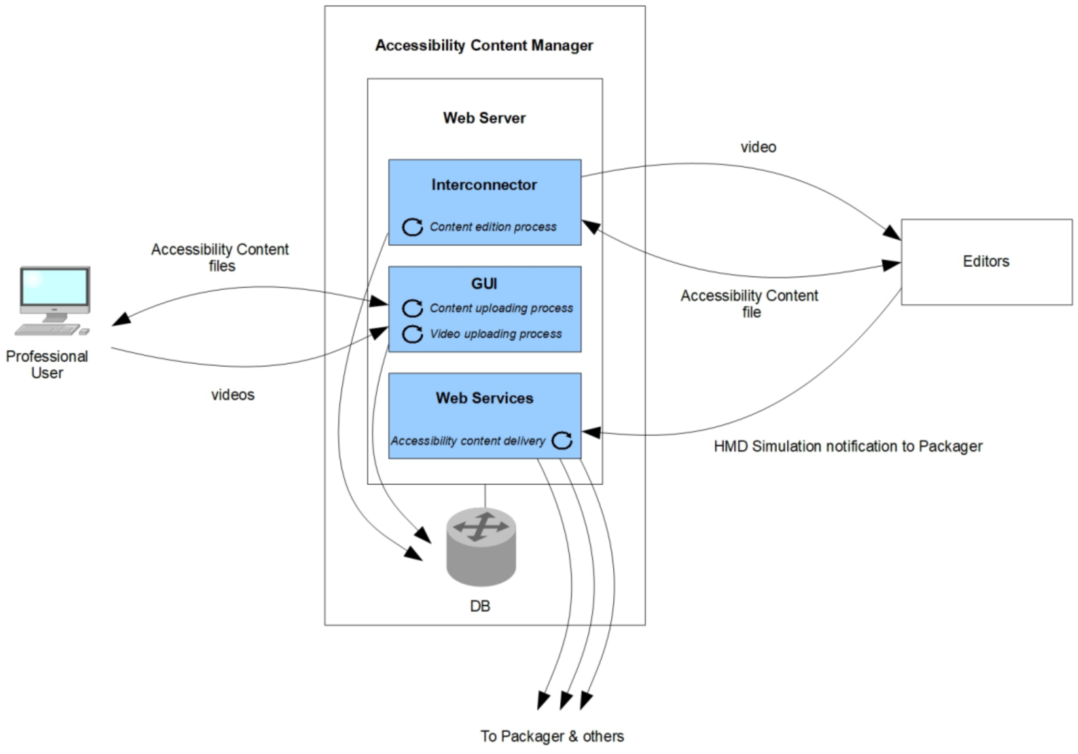
The object based audio editor runs as stand alone software on a Desktop PC.



Accessibility Content Manager
video
Object based audio Editor
OBA preview renderer
Editing functionalities
Audio input (various formats)
Object Based Audio Scene
HD

Figure 21- Deployment View Diagram - Object based audio renderer

3.4.2 Content Manager



Accessibility Content Manager
Web Server
Interconnector
Content edition process
GUI
Content uploading process
Video uploading process
Web Services
Accessibility content delivery
DB
Professional User
Accessibility Content files
videos
video
Editors
Accessibility Content file
HMD Simulation notification to Packager
To Packager & others

Figure 22- Deployment View Diagram - Accessibility Content Manager

3.4.3 Content Packager / Distribution

The following servers are required for the deployment of the content packager and distributer as implemented in T3.3:

- FTP for shared folder.
- Access to Amazon Cloud Server for audio rendering
- Video encoder and DASH packager
  - Offline for VoD or “fake” linear.
- Server for Web Service
- CDN

The schema in Figure 23 presents the implementation architecture of the version 1 of the server-side ImAc platform.

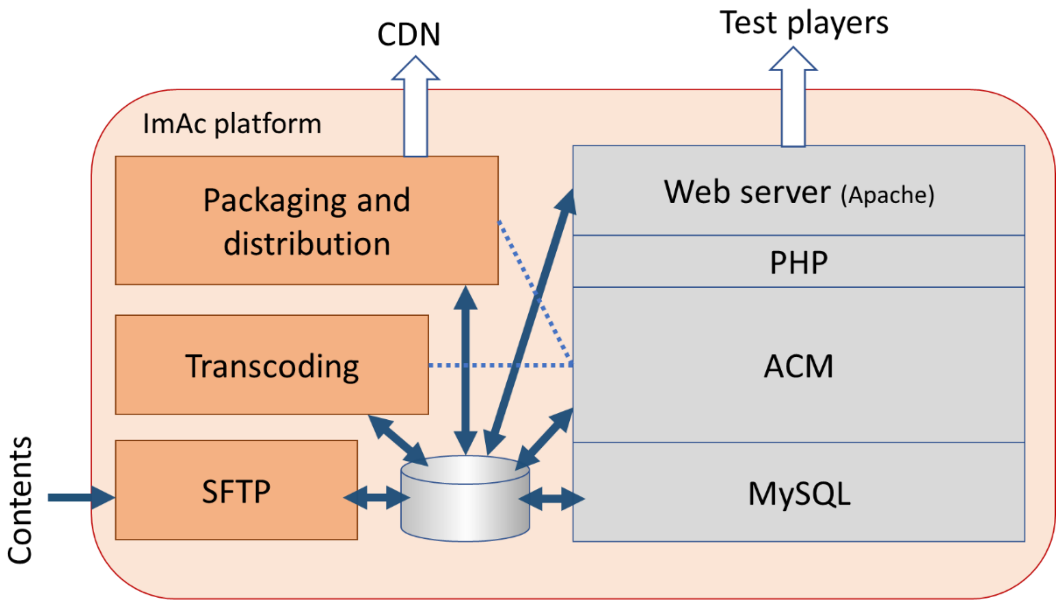


Figure 23: Platform implementation architecture for pilot 1

All components on the right (in gray) correspond to modules that work with the ACM, which are described in previous part. The other components (in orange) are described in this document with three main parts: ingest, transcoding, packaging and distribution.

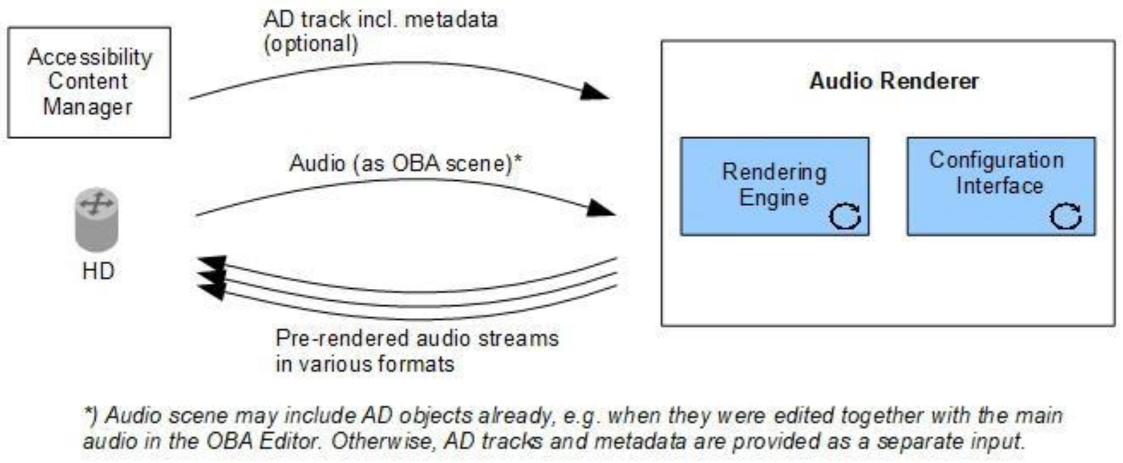

Accessibility Content Manager
AD track incl. metadata (optional)
Audio (as OBA scene)*
HD
Pre-rendered audio streams in various formats
Audio Renderer
Rendering Engine
Configuration Interface
*) Audio scene may include AD objects already, e.g. when they were edited together with the main audio in the OBA Editor. Otherwise, AD tracks and metadata are provided as a separate input.

Figure 24 - Deployment View Diagram - Audio Renderer

3.4.4 Player

Figure 21 illustrates the involved physical nodes or entities and their configurations on the consumer side. The considered scenarios can involve either only (maybe independent) companion devices (presenting both immersive and accessibility contents) or a single main TV (presenting traditional or immersive contents) and one or multiple (n) companion devices (presenting both immersive and accessibility contents).

The project contemplates two development phases. In the first one, only web scenarios will be considered. This means that both the main TV and companion devices will include a web-based player for the presentation of the ImAc media contents. In case that main and companion devices form part of a shared session, proper discovery, association and app launching mechanisms will be provided. These features are provided by the HbbTV standard. In addition, the DVB-CSS protocols, also adopted by HbbTV standard, will be used to guarantee synchronized playback processes across the involved devices. This is illustrated on the left part of Figure 21. The second development phase will only be considered given the availability of equipment supporting the latest version of HbbTV standard, v 2.0.x. This will allow synchronizing the playback of broadcast contents presented on the main TV with the playback of broadband related contents presented on the companion devices. In such a case, the discovery, association and app launching mechanisms are also natively supported by HbbTV 2.0.x, and thus being provided by the involved equipment / devices.

In these multi-screen scenarios, one device (typicall the main screen) will act as the master and the others (companion screens) as the slave. However, there is no restriction to enable interactive sessions in which the execution of playback control commands is shared between the involved devices.

The communications and interactions between the consumer devices and the Content Manager to be aware of the available contents, select them and retrieve them are shown in Figure 21.

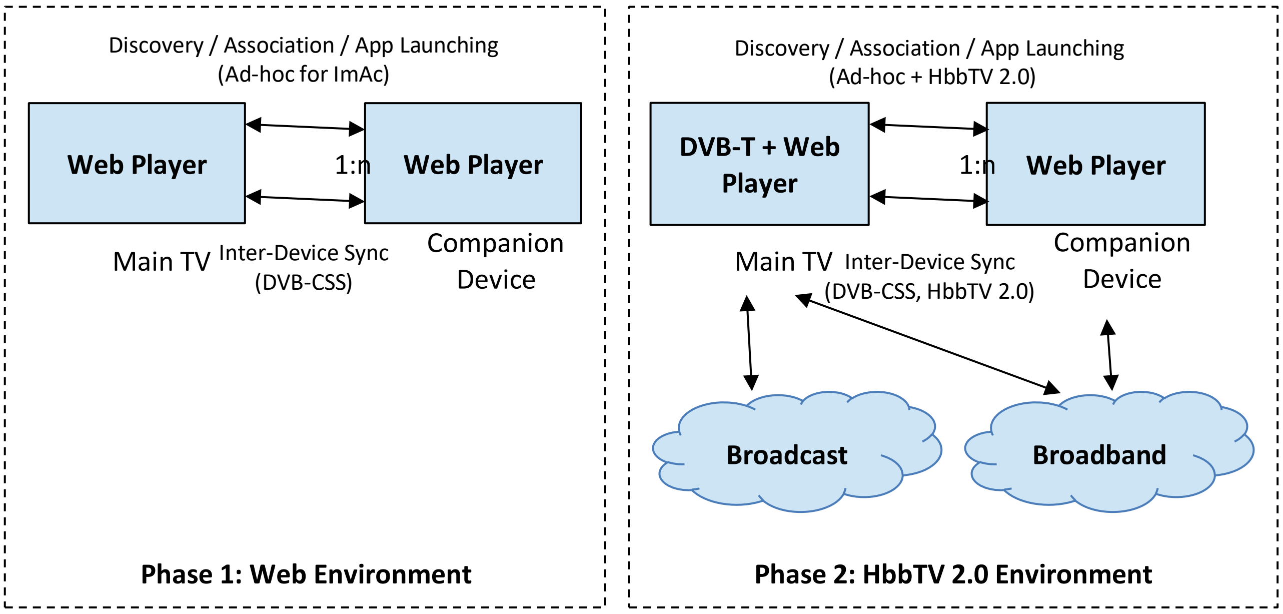


Figure 25 - Physical Nodes and configurations in the consumer side

3.4.5 Complete ImAc System and proposed workflow

The proposed workflow for the ImAc platform is shown in Figure 22. Each component within the workflow is described below indicating the how it is integrated with the ImAc platform, a description of the interface from phase 1, the final planned interface and testing details.

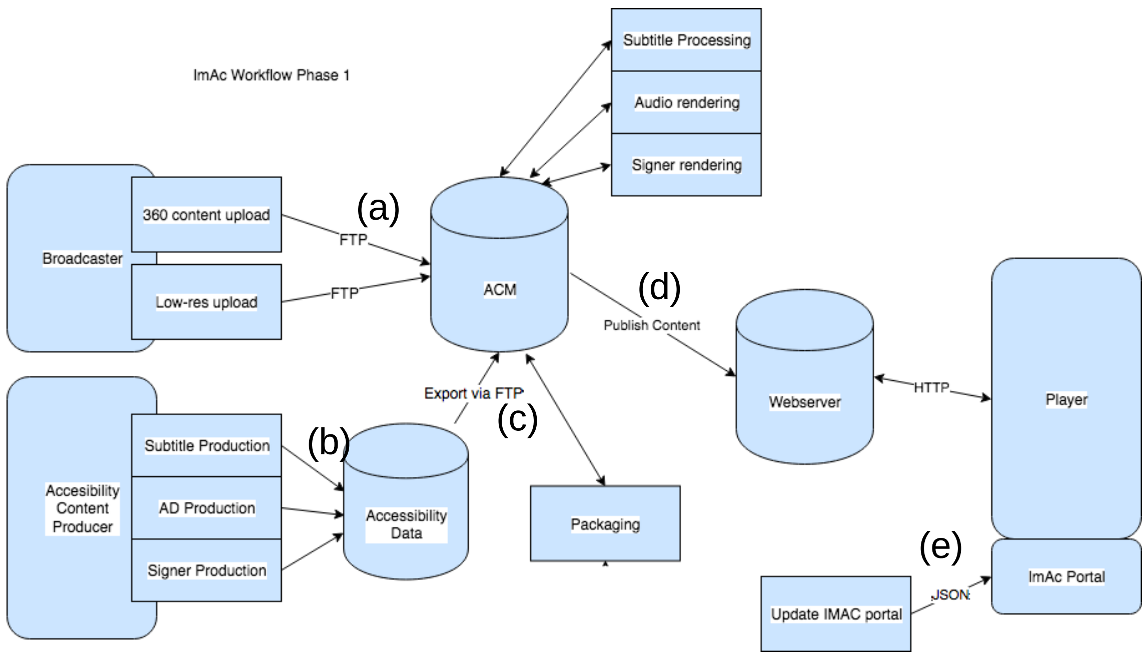


Figure 26 - Deployment View Diagram - complete ImAc system

360-degree content upload

Integration Point	Between the Broadcaster and ACM as shown in Figure 26(a)
Interface in Phase 1	Broadcaster (TVC, RBB) uploads HQ content in FTP. This FTP has a folder named “Input”. The name of the file uploaded is taken as the ID of the content and this ID will be used through the workflow.
Result	● HQ content is available on FTP server. ● Program ID is defined (by the file name). ● The upload is triggered by the upload of a signature file.
Planned Interface	A web client in the ACM
Testing Details	HQ content includes 360 video and one (or more) main audio mixes. Supported format(s) video: H.264 360º Equirectangular 4K or superior resolution Supported format(s) audio: AAC or WAV with FOA (= First Order Ambisonic), Binaural (stereo),

Preview (Low Resolution) Upload

Integration Point	Between the Broadcaster and ACM as shown in Figure 26(a)
Interface in	Following the content upload a low-res version is automatically generated and

Phase 1	stored within the ACM. The name of the file uploaded is the same as chosen in #1, but with the suffix "_lowres".
Result	● Low-res content is available in ACM. ● Program in ACM is linked to HQ content on ftp
Planned Interface	Automatically produced by the ACM
Testing Details	Low resolution content includes 360 video and one main audio mixes. Format video: mp4/h264 Format audio: aac, stereo

Subtitle Production

Integration Point	Between the Accessibility Content Producer and the Accessibility Content Database as shown in Figure 26(b)
Interface in Phase 1	Multiple steps: 1) Broadcaster (TVC, RBB) assigns subtitling work to one or more subtitlers from the CM interface (from the subtitling edit dialogue of the asset that appears on the right side of the screen when clicking the icon on the asset). 2) Subtitlers access to the ED interface (straight after logging to ACM when they only have subtitler permission or by clicking the ED icon from the CM interface when they also have permission on the CM interface). 3) Subtitler clicks on one of his/her pending subtitling works. 4) The web subtitling editor opens with the subtitling work and the corresponding video. The subtitler can create the subtitling and click the save button to save it to the ACM.
Result	● Subtitles are available in ACM.
Planned Interface	As Phase 1

AD Production

Integration Point	Between the Accesibility Content Producer and the Accessibility Content Database as shown in Figure 26(b)
Interface in Phase 1	Multiple steps: 1) Broadcaster (TVC, RBB) assigns AD work to audio descriptors from the CM interface. 2) Audio descriptors access to the ED interface (straight after logging to ACM when they only have audio descriptor permission or by clicking the ED icon from the CM interface when they also have permission on the CM interface). 3) Audio descriptor clicks on one of his/her pending AD works. 4) The web AD editor opens with the AD work and the corresponding video.

	5) The audio descriptor can create the AD and click the save button to save it to the ACM.
Result	● AD assets are available in ACM.
Planned Interface	As Phase 1

Signer Production

Integration Point	Between the Accessibility Content Producer and the Accessibility Content Database as shown in Figure 26(b)
Interface in Phase 1	Multiple steps: 1) Broadcaster (TVC, RBB) assigns SL work to signer from the CM interface. 2) Signer access to the ED interface (straight after logging to ACM when they only have signer permission or by clicking the ED icon from the CM interface when they also have permission on the CM interface). 3) Signer clicks on one of his/her pending SL works. 4) The web SL editor opens with the SL work and the corresponding video. 5) The signer can create the SL and click the save button to save it to the ACM.
Result	● Signer file package is available in ACM.
Planned Interface	As Phase 1

Accessibility data export to FTP

Integration Point	Between the Accessibility Database and the ACM as shown in Figure 26(c)
Interface in Phase 1	When all accessibility content was created, Broadcaster (TVC, RBB) goes to ACM and triggers export of all accessibility data to the FTP. ACM will take care of copying files and trigger conversion processes. There is a folder named “output” where all files shall be copied into. 1) Subtitle files are manually copied to the FTP. 2) For AD see 3.4.2 3) For Signer see 3.4.3
Result	● Accessibility files are available on FTP. ● Metadata about accessibility files is made available for packaging.
Planned Interface	As Phase 1

Audio Rendering

Integration Point	Between the Accessibility Database and the ACM as shown in Figure 26(c)
Interface in Phase 1	Multiple steps: 1) ACM (ANGLA) triggers audio rendering from ACM. 2) Audio renderer (IRT) processes audio. 3) Audio renderer copies pre-mixed audio streams (also main mix without AD) to FTP. There is a folder named "output" where the audio files shall be copied into.
Result	● pre-mixed AD audio streams are available on FTP.
Planned Interface	As Phase 1
Testing Details	Rendering can be triggered with HTTP-request. Renderer will get all AD audio snippets with accompanying metadata and the main audio mix in a zip-package. Json-file is used for setting renderer parameters (incl. output path, naming convention)

Signer Rendering / Processing

Integration Point	Between the Accessibility Database and the ACM as shown in Figure 26(c)
Interface in Phase 1	The signer video is manually uploaded to the server via FTP.
Result	● signer stream(s) is/are available on FTP in folder "output".
Planned Interface	The signer content will be automatically uploaded from the ACM

Publish Content

Integration Point	Between the ACM and content webserver as shown in Figure 26(d)
Interface in Phase 1	The publishing process is triggered in the ACM.
Result	● publishing process is triggered.
Planned Interface	This will be an automated process initiated from the ACM

Packaging

Integration Point	Between the Packager and ACM as shown in Figure 26(d)
Interface in Phase 1	From the metadata and assets provided, the packager (MSE) will manually segment and package all files and create an MPD file.
Result	● all files are packaged, MPD is created. Files are on the server, ready for playback.
Planned Interface	This will be an automated process initiated from the ACM

Update ImAc portal

Integration Point	Between the ACM and the ImAc portal as shown in Figure 26(e)
Interface in Phase 1	A json file listing all contents is created, so to the IMAC portal can include links to the MPDs and all required metadata.
Result	● program is published and can be played back via the IMAC-portal.
Planned Interface	This process will be automated

4. Dependencies

4.1. Player Dependencies

The player is built on a number of existing tools. These are detailed in Table 20 and shortly introduced in this section. More details can be found in D3.5.

Table 20 - Player Dependencies

Subtitles		IMSC.js
Video	2D	DASH.js
	360	Three.js (WebGL)
Audio	360	Omnitone.js (Ambisonics)
Sync		DVBCSS.js
Scene		SESSION_MANAGER.js
(Local) Inter Media Sync		intermedia_sync.js
Inter Device Sync		DVB-CSS.js

4.1.1 IMSC.js

https://github.com/sandflow/imscJS

ImscJS is a JavaScript library for rendering IMSC Text and Image Profile documents to HTML5. IMSC1 is a profile of TTML designed for subtitle and caption delivery worldwide.

4.1.2 DASH.js

https://github.com/Dash-Industry-Forum/dash.js/wiki

A reference client implementation for the playback of MPEG DASH contents via JavaScript and compliant browsers.

4.1.3 Three.js

Three.js is a cross-browser JavaScript library and Application Programming Interface (API) used to create and display animated 3D computer graphics in a web browser, without the need of proprietary plugins. Three.js uses WebGL. It is used to compose the 360º environments from traditional 2D videos processed by *DASH.js*.

4.1.4 Omnitone.js

https://github.com/GoogleChrome/omnitone

Omnitone is a robust implementation of ambisonic decoding and binaural rendering written in Web Audio API. Its rendering process is powered by the fast native features from Web Audio API (GainNode and Convolver), ensuring the optimum performance.

4.1.5 Intermedia_sync.jss

Library that will handle the temporal synchronization between the playback of all considered immersive and accessible contents within each single device.

4.1.6 DVBCSS.jss

Library with an implementation of the DVB-CSS (Companion Screens & Streams) protocol, adopted by HbbTV standard, to provide Inter-Device Synchronization (IDES) between a main TV and companion devices. In ImAc, this library will be also used to synchronize the playback between the involved devices.

4.1.7 SESSION_MANAGER.js

This library implements all the required functionalities for Session Management. It stores relevant information about the session, such as the metadata obtained from the Content Manager (e.g., describing the available contents, their relationships, etc.), the contents being currently presented, the available personalisation options together with the current settings, as well as the status of the session (e.g., elapsed time, duration, other active devices...).

5. Size and Performance

This architecture represents a proof-of-concept implementation and is not designed for a real, high performance broadcaster environment. Some components however will be scalable enough to be integrated in professional environments.

The architecture will however support the key sizing and timing requirements, as stipulated in the Platform Specification.

P1	Production Editors
P2	Accessibility Content Manager
P3	Content Packager and Distribution
P4	Player

5.1. Production Editors

Table 21 - Key sizing and timing requirements for the production editors

P1.1	Unlimited number of simultaneous web editors in remote stations (it depends only on the hardware capabilities of the server and authorized instances, not limited by software).
P1.2	Instant playback for previewing/verification, only a small buffer on the player when additional video has to be downloaded.
P1.3	Significant delay on the HMD verification playback, all the processes from the content packager have to be triggered.

5.2. Accessibility Content Manager

Table 22 - Key sizing and timing requirements for the accessibility content manager

P2.1	File and database capacity only limited by hard drive size of the server, not by software.
P2.2	Subtitling file upload/download: no limitation in size (they are manageable files) and fast to transfer.
P2.3	Audio description file upload/download: size only limited by Apache configuration (they can have considerable size), and some time is required.
P2.4	Sign language file upload/download: size limited only by Apache configuration (they are big files), and some time without closing the web browser is required.
P2.5	Video file upload/download: size limited by Apache configuration (they are quite big files), and quite a lot of time without closing the web browser is required.
P2.6	Metadata viewing/editing: fast and without limitation (only authorized users).

5.3. Content Packager and Distribution

Table 23 - Key sizing and timing requirements for the content packager and distribution

P3.1	Encoding: One simultaneous stream for main content.
P3.2	Packaging: One simultaneous stream.
P3.3	Delivery: CDN- unlimited number of players

P3.4	Delivery: Broadcast TS- unlimited number of players

5.4. Player

Table 24 - Key sizing and timing requirements for the player and shared sessions

P4.1	There is no reason to have multiple active players per consumer device. The software architecture and design of the ImAc player guarantee a proper presentation of the immersive and accessibility contents in a single player on current consumer devices (e.g. smartphones, tablets, HMDs...). It is true even considering the use of web-based technologies, and the limited hardware and software resources of the current consumer devices.
P4.2	The designed solutions to achieve a proper association between the involved devices and a synchronized playback in the session in local scenarios are scalable and lightweight. Thus, the number of simultaneous devices presenting ImAc media contents in local scenarios will be mostly limited by the available bandwidth (e.g. determined by the contract with the Internet Service Provider (ISP).
P4.3	The total number of involved players, in all the active distributed scenarios, will be determined by the scalability of the service provider / broadcaster resources (e.g., Content Manager, CDNs, number of servers...). A proper dimensioning of these resources, together will proper strategies to distribute/balance them will maximize the scalability in terms of number of concurrently ImAc players and sessions.

6. CONCLUSIONS

This document has provided a comprehensive architectural overview of the ImAc project, using a number of different architectural views to depict key aspects of the system. It is intended to capture and convey the significant architectural decisions, which have been made on the system based on User requirements (D2.2), and Platform Specifications (D2.3) defined in WP2.

In Chapter 2 we have described what we need to do. This is based on the user requirements gathered in D2.3. Chapter 3 explained how we are going to do it and provides a blueprint for the ImAc system. Chapter 4 details the tools we are going to use to make the platform work. Finally Chapter 5 describes the size and performance requirements in order for success.

This document also provides the basis for the development work in WP3 and WP4. Specifically it feeds into the following tasks:

- Immersive Platform
    - T3.2 Content formats and storage
    - T3.3 Content Packaging and distribution
    - T3.4 Accessibility Interface
    - T3.5 Player
    - T3.6 Integration and Testing
- Accessibility Service
    - T4.1 Subtitling Services
    - T4.2 Enhanced Audio Services
    - T4.3 Sign Language

Section 2.1 (Key Requirements), 3.1 (Scenarios) and 5 (Size and Performance) also form the key metrics for D3.6 (Integration and Testing Report), scheduled for an final release in M20.

REFERENCES

[1] Kruchten, Philippe (1995, November). Architectural Blueprints — The “4+1” View Model of Software Architecture. IEEE Software 12 (6), pp. 42-50.

[2] Hughes, CJ , Armstrong, M, Jones, R and Crabb, M 2015, Responsive design for personalised subtitles, in: The 12th Web for All Conference, 18-20 May 2015

<END OF DOCUMENT>