Traction
Opera co-creation
for a social
transformation

D2.4: Intermediate TRACTION Toolset

DISSEMINATION LEVEL	PUBLIC (PU)
WORK PACKAGE	WP2 TOOLSET DESIGN AND DEVELOPMENT
DELIVERABLE NUMBER	D2.4
VERSION	V4.0
SUBMISSION DATE	30/04/2021
DUE DATE	30/04/2021

This Project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 870610

Authors

Authors in alphabetical order		
Name	**Organisation**	**Email**
Alina Striner	CWI	alina.striner@cwi.nl
Nacho Reimat	CWI	nacho.reimat@cwi.nl
Pablo Cesar	CWI	p.s.cesar@cwi.nl
Thomas Röggla	CWI	t.roggla@cwi.nl
Anderson Simiscuka	DCU	anderson.simiscuka2@mail.dcu.ie
Mohammed Amine Togou	DCU	mohammedamine.togou@dcu.ie
Rohit Verma	DCU	rohit.verma@dcu.ie
Ana Dominguez	VICOM	adominguez@vicomtech.org
Hector Rivas	VICOM	hrivas@vicomtech.org
Iñigo Tamayo	VICOM	itamayo@vicomtech.org
Mikel Zorrilla	VICOM	mzorrilla@vicomthech.org
Stefano Masneri	VICOM	smasneri@vicomtech.org

Control sheet

Version history			
Version	**Date**	**Modified by**	**Summary of changes**
0.1	24/02/2021	Anderson Simiscuka	First ToC
0.2	31/03/2021	Stefano Masneri	Co-Creation Stage
0.3	01/04/2021	Mohammed Amine Togou	Non-Real Time Streaming Adaptation
0.4	01/04/2021	Anderson Simiscuka	Immersive Media and Real Time Adaptation
0.5	09/04/2021	Thomas Röggla	Added Co-Creation Space features section
0.6	10/04/2021	Thomas Röggla	Added Co-Creation Space architecture section
1.0	12/04/2021	Anderson Simiscuka	Added content from Alina related to the Co-Creation Space
1.1	13/04/2021	Nacho Reimat, Alina Striner	Added SocialVR sections
1.5	16/04/2021	Pablo Cesar	Review of the initial draft
1.6	16/04/2021	Thomas Röggla	Additional figures and text in section 4.1.3
1.7	20/04/2021	Anderson Simiscuka	Executive Summary, Introduction, Conclusion and updated sections on Accessible 360° Player
1.8	21/04/2021	Stefano Masneri	Modify section on Co-Creation Stage
2.0	22/04/2021	Anderson Simiscuka	Version before review
3.1	26/04/2021	Thomas Röggla	Addressed review comments in sections 3.1 and 4.1
3.2	26/04/2021	Stefano Masneri	Addressed review comments
4.0	27/04/2021	Anderson Simiscuka	Addressed review comments

Peer review		
	Reviewer name	**Date**
Reviewer 1	Pilar Orero	23/04/2021
Reviewer 2	Irene Calvis	27/04/2021

Legal disclaimer

Table of Contents

List of Figures

Abbreviations

Abbreviation	Definition
AWS	Amazon Web Services
CSV	Comma-separated values
DASH	Dynamic Adaptive Streaming over HTTP
DoA	Description of Action
EC	European Commission
ERD	Entry Relationship Diagram
FG	Focus Group
GA	General Assembly
HLS	HTTP Live Streaming
HTTP	Hypertext Transfer Protocol
SQL	Structured Query Language
WP	Work Package
WPL	Work Package Leader

Executive summary

This deliverable describes the intermediate prototype implementation of the TRACTION project toolset, after sixteen months into the project. It is a demonstrator type deliverable and this report shows the evidences of the software release that has been implemented. This is the second iteration of this deliverable, presenting the features implemented after the pilot activities of the first year of the project. These features include improvements fixing issues identified on feedback from users and new functionalities based on requirements gathered during focus groups and open pilots. The implementation of the technologies is part of the WP2 activities.

This deliverable is divided into four main sections.

Section 2 presents a summary of the current state of the toolset: The Co-Creation Space, the Co-Creation Stage and the Immersive Experience Engine, which consists of the Social VR technology and the Accessible 360° Player.

Section 3 details the new features and contributions of the toolset based on the requirements from the pilot activities of the first year. Issues identified and corrected are also presented. This section also presents user interfaces and new functionalities of the toolset.

Section 4 focuses on the updated architectures of the tools, which have been extended to support the new features of the intermediate toolset. Diagrams illustrating the production environments, major architectural components and database models are also presented.

Finally, Section 5 details the plans for the next pilot activities, focus groups and evaluation sessions for requirements gathering.

The main contributions of this deliverable include:

- A description of the current state of the second iteration of the toolset;
- The new functionalities, requirements and the updated architecture of the second iteration of the toolset;
- The plans for the next reporting period, including pilot activities and sessions for user testing.

The next iteration of this deliverable will include the final implementation of the TRACTION toolset, based on the new requirements and feedback from the next wave of pilot activities and workshops with communities, artists, and audiences.

1 Introduction

1.1 Purpose of the deliverable

This deliverable presents the updates to the toolset architecture, the new features implemented after sixteen months into the project, as well as the details on pilot activities, gathered requirements and plans for year 2. The look and feel of the tools and the architectural design are also presented with screenshots and diagrams.

The second iteration of the toolset development focuses on improving the user interfaces based on pilot activities and user feedback. New functionalities have also been implemented according to trial needs and requirements gathered during year 1. Some tools have also received a new name towards a more descriptive term taking advantage of the functionality. The main components of the project's toolset include:

- The Co-Creation Space, described in previous deliverables as the MediaVault, is a web-based tool which provides features for video transcoding, authentication, content upload, mobile support, automatic translation, database support, users' collaboration, content tagging and commenting;
- The Co-Creation Stage, described in previous deliverables as the Performance Engine, is a web-based tool that provides multi-screen capabilities, remote users participation and viewing, delivery of real-time content, customisable templates for shows, support to pre-recorded content, and content orchestration;
- The Immersive Experience Engine, consisting of the Social VR technology and the Accessible 360° Player. The Social VR solution allows multiple users to meet, interact and playback immersive contents together in a common virtual space using Virtual Reality headsets. The accessible 360° player supports 360° videos playback with immersive audio, accessible features, and is used for the delivery of pre-recorded immersive footage of operas along with accessibility services.

1.2 Intended audience

This deliverable audience is anyone interested to learn more about the technologies that enable the achievement of the goals of the TRACTION project, including its aim of using Opera as a vehicle for facilitating social inclusion and the development of tools for artistic creation, community participation and immersive opera distribution. In particular, this deliverable focuses on the technological innovations of the project hence the audience includes media producers, co-creation researchers, software developers, Opera enthusiasts, etc.

1.3 Structure

This deliverable is divided into four main sections. Section 2 presents a summary of the current state of the toolset. Section 3 details the new features and contributions of the toolset based on the requirements from the pilot activities of the first year. Section 4 focuses on the updated architectures of the tools, which have been extended to support the new features of the intermediate toolset. Finally, Section 5 details the plans for the next pilot activities, focus groups and evaluation sessions for requirements gathering.

2 Toolset – A Summary

This section presents a description of the current state of the toolset focusing on the goal of each tool in the trials of the TRACTION project. There is also information on how the toolset has been used so far in pilot activities and the requirements gathered in the first year based on feedback and focus groups.

Two of the tools of the toolset, the MediaVault and the Performance Engine have been re-branded as Co-Creation Space and Co-Creation Stage, respectively. New logos have also been created to reflect the functionality of each tool, as seen in Figure 1. The tools related to virtual reality and immersive content are the Social VR component and the Accessible 360° Player, within the Immersive Experience Engine.

Figure 1: Co-Creation Space and Co-Creation Stage logos.

The Co-Creation Space logo represents the collaboration between different participants towards the co-creation, and the Co-Creation Stage logo is represented by multiple co-located stages performing together.

2.1 Co-Creation Space

The Co-Creation Space is a flexible web-based tool that allows for sharing, communication, and collaboration between professional and community participants around media objects. Created through a research-through-design user driven process, the Co-Creation Space bridges the gap between social sharing and active collaborative authoring between community members by supporting a grassroots community production process.

Using the Co-Creation Space, users can create an account, upload a profile image, and select topic interests to follow. As part of the collaboration process, users can create a new post, and upload images and videos in a variety of formats that will be displayed within the tool, and that can be translated into several languages. Created posts are summarized in a text and media timeline view by topic. Users can browse through posts by topic, or search for posts using specific tags or keywords. Users can then comment on images and videos using text, emojis, and respond with image and video media. Users receive notifications about responses to posts they've created, allowing them to continue the collaboration process. Figure 2 provides a conceptual diagram of the capabilities of the tool, and Figure 3 contains a use-case diagram, illustrating the collaboration process.

Figure 2: The Co-Creation Space as a tool enabling distributing remote communication around media objects.

Figure 3: Co-Creation Space Use-Case Diagram

In Deliverable D2.3 we presented the user-centred design process we followed to first gather requirements, and then to evaluate user experience of our system. We gathered requirements with the three groups of trial leaders. After designing the tool, we conducted a pilot activity with LICEU trial users in Barcelona, Spain that included Sínia creatives and students from Escola Massana. During the study, participants tested the usability and accessibility of the interface, and responded to interview questions about the value of the Co-Creation Space as a communication and co-creation tool.

After the first year, we identified three main issues that need to be addressed as requirements, gathered in D2.3. These include interface changes, image and video processing features, and the design of a meaningful timeline summary.

Interface: Study participants gave thorough feedback about the visual design of the interface, menu, search, and tagging capabilities, and about the user profiles. This included changing the colours and visual layout, making the menu and tagging features clearer, and adding more functionality to the search feature and user profiles. In the timeline view, participants expressed the desire to search by words, dates, users, and types of content.

Image Processing: Participants expressed a desire to trim images and videos, to mark them up, and to preview videos before uploading them. In addition, some wanted to change the contrast, brightness, white balance and saturation, whereas others wanted to modify the light of the image but not the contrast. Participants also wanted preview, compression, and comment features for video editing. This included being able to watch video previews, adding comments to a video, and compressing it before uploading.

Timeline Summary: When asked about an auto-generated summary, participants wanted a way to show their working progress about work already done in order to follow the evolution of work.

2.2 Co-Creation Stage

The Co-Creation Stage (formerly known as Performance Engine) is a web-based tool that enables distributed performances connecting different stages and people. Using the Co-Creation Stage tool, the artists can create a template for their show, defining beforehand a number of scenes, the number of stages, the screens/displays/projectors/devices at each stage and audio-visual assets including live and pre-recorded content.

During the show, thanks to a specific Web interface, an operator has three choices a) is able to manage the transition from one predefined scene to the next one or modifying on-the-fly (if needed) all the settings according to the requirements of the performance. b) The operator can also orchestrate the content to be shown on each output display, no matter if it is a screen on a stage or any display device used by a remote viewer or participant. Finally, c) the operator can also modify the layout template of each output display, providing the desired combination of assets on each screen. Figure 4 provides a conceptual diagram of the capabilities of the tool.

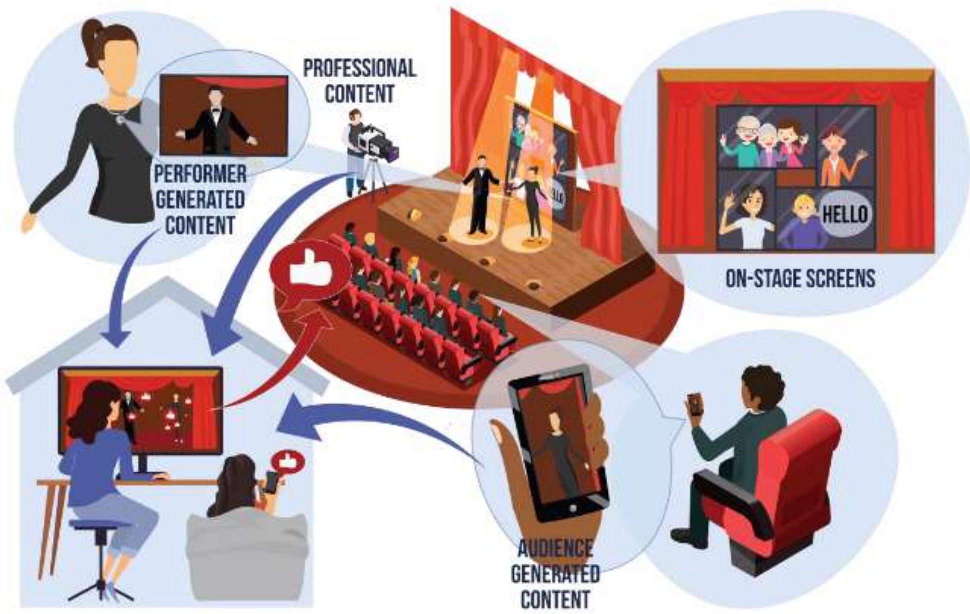


Figure 4: The Co-Creation Stage as a tool enabling distributed performances.

In Deliverable D2.3 we presented the user-centred design process followed to gather its requirements and provide the architectural design of the first release implementation. The Co-Creation Stage has been tested by the SAMP team in Leiria. There, we simulated a performance to evaluate the technical capabilities of the tests and conducted user tests to identify the problems and possible improvements of the UI and UX of the tool. After the first year, we identified three main issues that has been addressed as requirements for the second year of the project. The three issues are summarized here:

Role of the director and role of the operator: we have to check if there is a need to differentiate between the role of the director, who creates the whole story, and the role of the operator, who creates the template for the shows and then manages it during the performance.

Universality vs. Professional tool: based on the tests in Leiria we identified two different scenarios. On one hand we could try creating a tool that can be used in any browser and any device, with a UI as simple as possible and a UX that easily let people join the show, maximising the co-creation capabilities. On the other hand, a tool that allows using professional video and audio hardware and focuses on creating high quality content would be better in term of artistic value.

Deterioration of audio quality: this is a more technical issue which stems from the fact that the technology used by the Co-Creation Stage, the *getUserMedia()* function from the Stream API, has been developed for the use case of videocalls, not for live music streaming. The browsers apply several audio filters to improve voice communication but severely impacting the transmission of music.

2.3 Immersive Experience Engine

The Immersive Experience Engine of TRACTION contains two main tools: The Social VR component and the Accessible Immersive 360° Player.

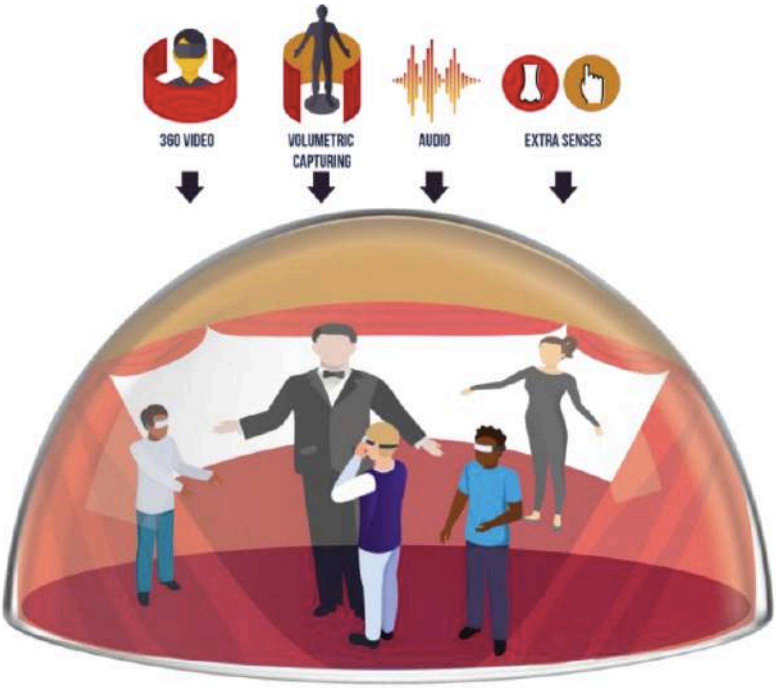


Figure 5: The Immersive Experience Engine supporting communication, collaboration and content viewing in a virtual space.

The Social VR tool will be used to enhance the virtual opera, in conjunction with the other tools of the toolset. It focuses on enriching the main VR performance by adding a virtual space for co-creation, communication, collaboration and audience interaction. The accessible immersive 360° player offers support to the playback of 360° videos with immersive audio (ambisonics) and is used for the delivery of pre-recorded immersive footage of operas.

2.3.1 Social VR

During the first period, INO and CWI conducted three focus groups - Immersive Experience Engine Focus Group (FG2), reported in D4.2 – for better understanding the potential that VR holds for Opera. Based on the results, we have decided to explore the benefits and drawbacks of social VR technology for novel Opera experience. In particular, we intend to re-use and extend the solution implemented in the H2020 VRTogether project by CWI and other partners (https://vrtogether.eu).

This solution would allow multiple users to meet, interact and playback immersive contents together in a common virtual space using Virtual Reality headsets. The self-representation of the users could vary from a 3D avatar to a real-time captured point cloud, which would provide a photorealistic representation of the users.

2.3.2 Accessible 360° Player

The accessible 360° video player, part of TRACTION's Immersive Experience Engine, offers support to the playback of 360° videos with immersive audio (ambisonics) and is used for the delivery of pre-recorded immersive footage of operas. As described in deliverable 2.3, the web-based player is based on the ImAc player from the H2020 project Immersive Accessibility[1]. Figure 6 presents an example of a user story in this context in presented below:

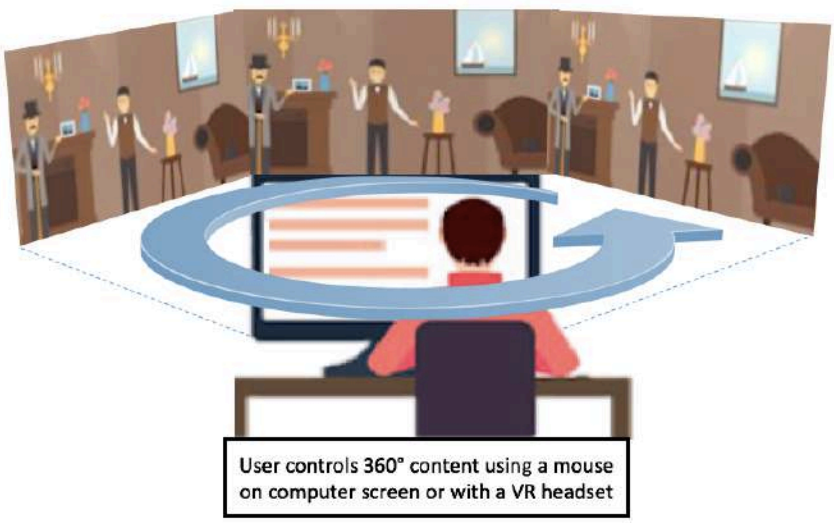


Figure 6: 360° video navigation.

Users can navigate through 360° videos using a mouse or a VR headset, in a web browser. Users can also perceive the audio direction as they move, if the footage is recorded with ambisonics technology. The player supports manual subtitles, sign language, audio description and audio subtitles (when tracks/files are available).

A number of improvements were already added to the accessible player during year 1, so it now supports WebXR and has a new interface in line with TRACTION's visual language. The player has also been extended to support quizzes and feedback forms for the user experience evaluation. This feature was used on a test for evaluating user perceptions on immersive audio and video.

The main use of the player is the playback and dissemination of new and existing 360° opera content.

[1] https://cordis.europa.eu/project/id/761974

3 Description of the Toolset New Functionalities

This section focuses on the new features and contributions of the toolset since the last demonstrator deliverable – based on the latest requirements from last year's pilot actions.

3.1 Co-Creation Space Features

Through the feedback gathered from the pilot activities conducted in December 2020 in Barcelona, the team identified a series of basic improvements to the UI and its functionality. Moreover, several partners enquired about the possibility of a more open, lengthier trial. During this extended trial period, the Co-Creation Space was to be deployed into a stable production environment and fitted with functionality for tracking and analysing user behaviour to gain insight into how trialists use the Co-Creation Space. Finally, issues and requirements still open from the previous iteration were triaged, i.e. the team assessed which open issues were still relevant, which issues needed to be reformulated and/or adapted and which issues were not relevant anymore and could be closed.

The activities outlined in the preceding section were divided into three phases, targeted at the immediate future, the preparation for the suggested open trials and the longer term, i.e. the time until the second year pilot activities. The following sections describe each of these phases in greater detail.

3.1.1 Pilot & Partner Feedback

During and after the pilot activities held in Barcelona in December, trialists provided the team with good insight into how people use the UI of the Co-Creation space and allowed us to identify bugs and areas where the user interface was inconsistent or did not work correctly. Moreover, they provided direct feedback on how the application could be improved. Part of this feedback fed into the requirement gathering process for the second iteration and associated pilot activities and other parts were logged as issues in the application's issue tracker to be addressed immediately. These latter issues mainly consisted of small bugs and improvements to the user interface or functionality which was straightforward to implement.

Table 1: Features and bugs identified during the pilot activities or from feedback by partners

Type	Description	Priority	Effort
Feature	Logging of post filtering	Medium	Medium
Feature	Filter posts by user group	High	Low
Feature	Ability to edit tags when editing post	Low	Medium
Feature	Resize profile pictures	Medium	Medium
Feature	Notifications on comments to posts	Medium	High
Feature	Rendering of files in comments	Medium	Medium
Feature	Editing of media in post	Medium	Medium
Feature	Add participant code during signup	Medium	High

Feature	Automatic language identification during transcription	Low	Medium
Feature	Password reset feature	Medium	High
Feature	Render subtitles in audio player	Low	Medium
Bug	Prevent post form submission when hitting enter key	Medium	Low
Feature	Prompt user before deleting post	Low	Low
Bug	Username breaks layout if it is too long	Medium	Low
Feature	Ability to upload files of arbitrary type	Medium	Medium
Feature	Scope interest topics to user groups	Low	Medium
Bug	Video playback does not work on iOS	High	Medium
Bug	Check post owner before deleting post	Medium	Low
Feature	Ability to change user group	Medium	Low
Feature	Display username in profile	Medium	Low
Feature	Store user-agent with internal navigation	Medium	Low

Table 1 lists the feature requests suggested and bugs discovered during the first pilot activities in December and feedback gathered from project partners. These issues were inserted into the project's issue tracker and most low-effort issues were addressed immediately. Bigger feature requests were addressed during implementation work done in preparation for the open trials.

3.1.2 Issue Triage

All issues left unaddressed during the first iteration were triaged and most of them had either been implemented during the implementation of other issues, were marked as obsolete or re-evaluated and assigned new milestones. A total of 11 issues were affected during this triage process. One of which was the face-recognition functionality for uploaded media items, which was redefined as a separate service and assigned to the issue backlog for the second iteration of the application.

Table 2: Issues affected by the triage and the triage results

Issue	Triage Result
Setting to change between global and timestamped comment	Reassigned
Generate thumbnails for images	Reassigned
Move emoji-reactions from post to multimedia	Reassigned
Video face detection	Postponed
Handle form responses	Reassigned
Improve VTT generation algorithm	Postponed
Switch between audio tracks	Won't fix
Add multiple audio tracks to video	Won't fix
Listen to uploaded audio	Reassigned
Extract async actions to separate table	Reassigned
Download subtitles and captions	Won't fix

Table 2 shows the issues affected by the issue triage and the result of the triage process. Issues marked with `Won't fix` have become obsolete and were closed, whereas issues marked with 'Postponed' were put in the issue backlog and will be addressed at a later stage. Finally, issues marked with `Reassigned` were assigned a new milestone and implemented.

Part of the issue triage process was also devoted to updating the application's internal documentation and improvement of the unit test suite. While the number of unit tests was already comprehensive as far as the core application code was concerned, it was somewhat lacking in the area of tests for the database abstraction models. Therefore, during this phase, we ensured that all model classes were covered by unit tests and the internal documentation was brought up to speed. Figure 7 shows the unit test coverage report for the database model classes.

```
------------------------------|---------|----------|---------|---------|-------------------
File                          | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
------------------------------|---------|----------|---------|---------|-------------------
All files                     |   84.19 |     9.38 |      75 |   84.16 |
 models                       |   99.73 |    66.67 |     100 |   99.72 |
  associations.ts             |     100 |      100 |     100 |     100 |
  async_job.ts                |     100 |      100 |     100 |     100 |
  audio_content.ts            |     100 |      100 |     100 |     100 |
  data_container.ts           |     100 |      100 |     100 |     100 |
  emoji_reactions.ts          |     100 |      100 |     100 |     100 |
  index.ts                    |     100 |      100 |     100 |     100 |
  interest.ts                 |     100 |      100 |     100 |     100 |
  internal_navigation_step.ts |     100 |      100 |     100 |     100 |
  likes.ts                    |     100 |      100 |     100 |     100 |
  metadata.ts                 |     100 |      100 |     100 |     100 |
  multimedia.ts               |     100 |      100 |     100 |     100 |
  multimedia_interaction.ts   |     100 |      100 |     100 |     100 |
  notifications.ts            |     100 |      100 |     100 |     100 |
  permissions.ts              |     100 |      100 |     100 |     100 |
  post.ts                     |   95.65 |       50 |     100 |   95.45 | 184
  post_references.ts          |     100 |      100 |     100 |     100 |
  preferences.ts              |     100 |      100 |     100 |     100 |
  search_query.ts             |     100 |      100 |     100 |     100 |
  subtitles.ts                |     100 |      100 |     100 |     100 |
  tag.ts                      |     100 |      100 |     100 |     100 |
  tag_references.ts           |     100 |      100 |     100 |     100 |
  thread.ts                   |     100 |      100 |     100 |     100 |
  topic.ts                    |     100 |      100 |     100 |     100 |
  user_group.ts               |     100 |      100 |     100 |     100 |
  user_references.ts          |     100 |      100 |     100 |     100 |
  users.ts                    |     100 |      100 |     100 |     100 |
 util                         |   24.74 |     3.45 |    7.14 |   23.33 |
  index.ts                    |   24.74 |     3.45 |    7.14 |   23.33 | 21-32,49,64-70,85-98,110-114,123-131,141-158,166,176-186,197-202,233-255,270-280
------------------------------|---------|----------|---------|---------|-------------------
Test Suites: 23 passed, 23 total
Tests:       125 passed, 125 total
Snapshots:   0 total
Time:        8.2 s
Ran all test suites matching /models/i.
```

Figure 7: Coverage report for model unit test suite

3.1.3 Preparation for the Open Trial

During the pilot activities in December the involved partners expressed interest in using the platform with a wider circle of users and for an extended amount of time. Later, also other creative partners in the project expressed such interest. For such an extended open trial, the team decided to take further feedback from the creative partners into account. Moreover, we wanted to gain deeper insight into how users during such an open trial would use the application and therefore had to implement a series of features which allowed us to track and analyse user behaviour. All these issues were collated into a new milestone in the project's issue tracker, assigned projected priorities and effort levels and implemented in the lead up to the start of the open trials. Given the fact that the creative partners in the project were located in different countries, the user interface also needed to be translated into additional languages. These additional languages were:

- Irish
- Portuguese

The translations were compiled by internal project partners and can be changed dynamically by the users of the application according to their preference.

Finally, in preparation for the open trials, a number of features to track and analyse user behaviour had to be implemented. Table 3 lists issues that were identified and subsequently implemented during the preparation phase for the open trials.

Table 3: Issues addressed in preparation for the open trials

Type	Description	Priority	Effort
Feature	Setting to change between global comment and timestamped comment	Low	Medium
Feature	Listen to uploaded audio	High	High
Feature	Replace video.js with DASH.js	High	High
Bug	Post description should be optional	Medium	Low
Bug	Skip profile picture upload during signup	Medium	Low
Bug	Change username icon in signup form	Medium	Low
Bug	Only be notified about posts in group	Medium	Medium
Feature	Ability to download media	High	Medium
Bug	Hide title when editing a comment	Medium	Low
Feature	Join group on signup	High	Medium
Feature	Add post without media	Medium	Low
Feature	Deleting of posts/comments	High	Low
Feature	Removal of /id prefixed on routes	Low	Low
Feature	Move emoji-reactions from post to multimedia	Medium	Medium
Feature	Generate image thumbnails	Medium	Medium
Feature	Create stable production deployment environment	High	High
Feature	Internal navigation tracking	Medium	High
Feature	Tracking of search queries	High	Medium
Feature	Tracking of media interactions	High	Medium
Feature	Support for user groups	High	High
Feature	View counter for media items	High	Medium
Feature	Rename video router	Low	Low
Feature	Integrate Google Analytics	High	Low

3.1.4 Summary of new features

In this section, the new key features that were implemented since the pilot activities will be described and illustrated using screenshots from the running production environment that is being used in the open trials right now.

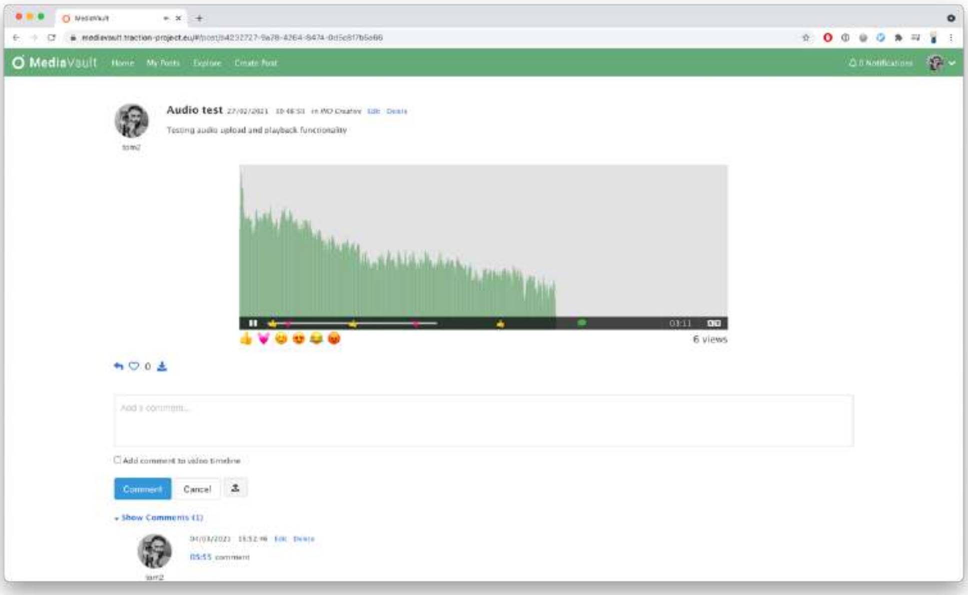


Figure 8: Audio playback with spectrum analyser

Figure 8 displays the new audio playback feature, which is implemented as a custom audio player supporting streaming of DASH and HLS content. Alongside audio playback, the player is able to render subtitles and displays a live spectrum analyser.

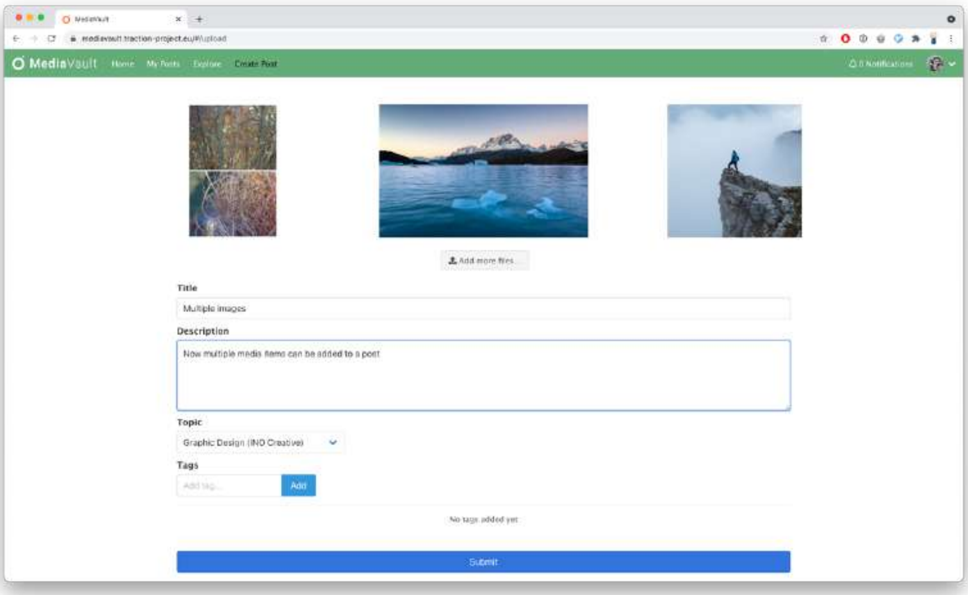


Figure 9: Multiple file upload during post creation

Figure 9 shows how a user can upload multiple media items of arbitrary type during post creation and moreover, Figure 10 shows how such a post is displayed once it has been created. Finally, Figure 11 shows multiple file support during editing of a post. Note that in this screen files can now also be deleted from a post. Adding and deleting of tags has also been implemented in the post editing component.

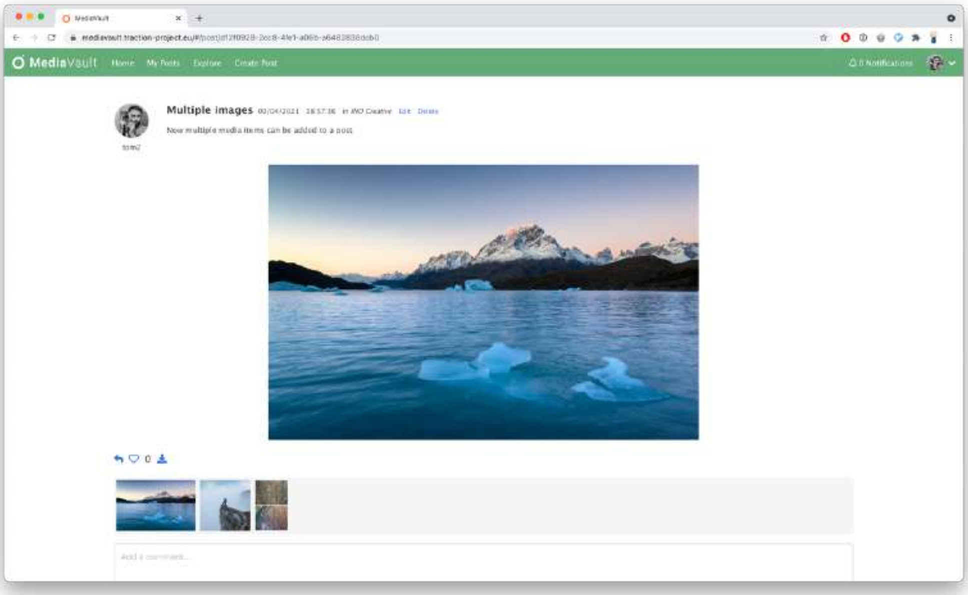


Figure 10: Viewing a post with multiple media items

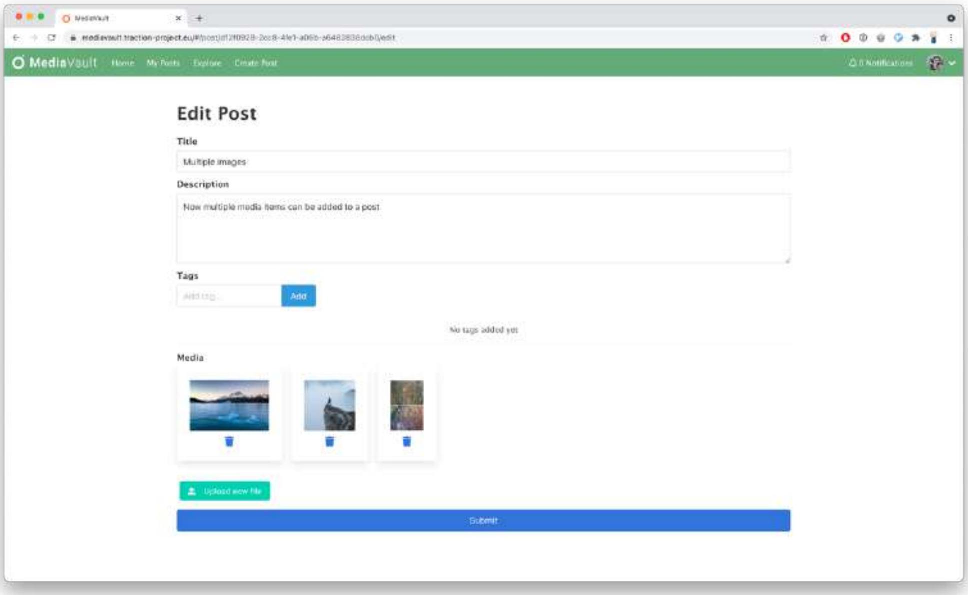


Figure 11: Support for multiple files and tags when editing a post

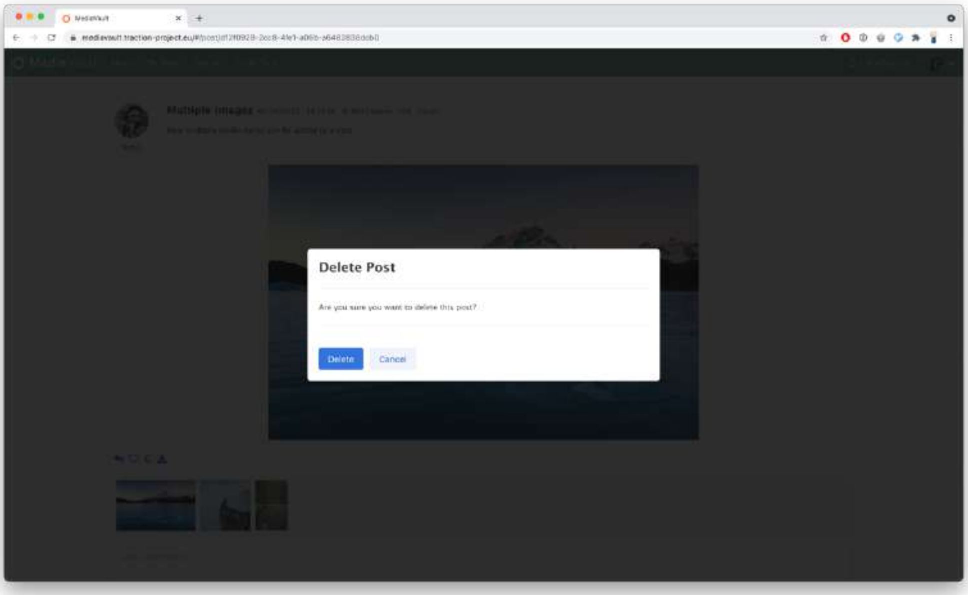


Figure 12: Prompting user for confirmation before deleting post

Figure 12 illustrates the prompt that asks the user to confirm the deletion before a post is actually deleted.

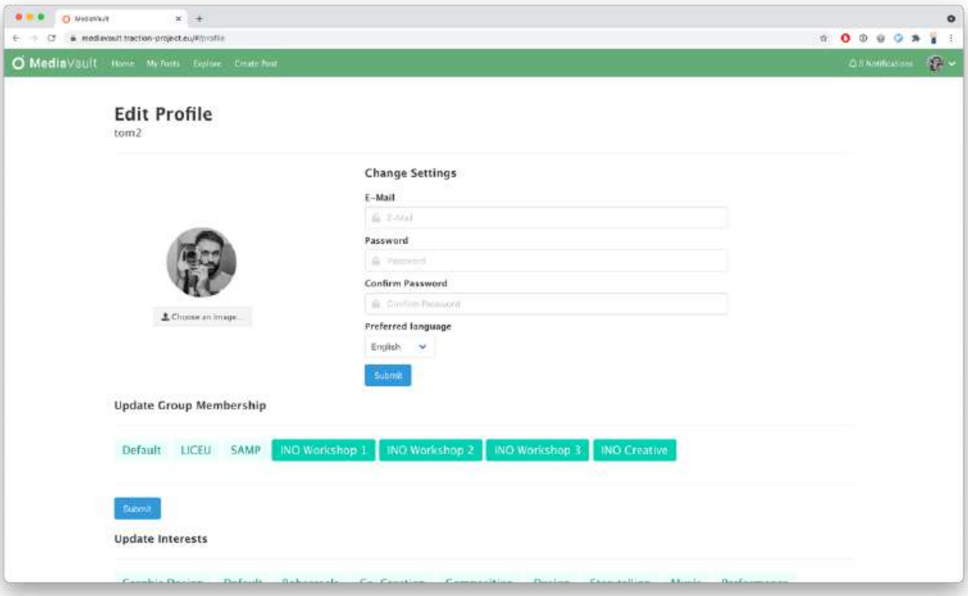


Figure 13: New profile editing page

Figure 13 shows the updated profile editing page, with added interface for updating group membership. Moreover, a field for inserting an e-mail address was added, which when filled in allows the user to reset their password in case they forgot it. Figure 14 shows the interface where a user can request a password reset. If the given e-mail address is registered with the system, a message with a reset link is sent to it.

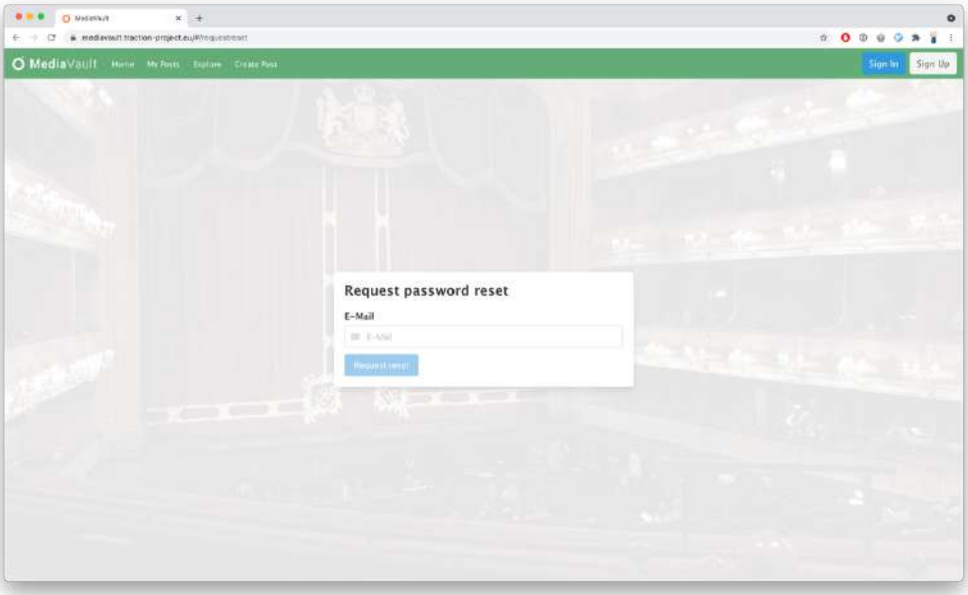


Figure 14: Interface for requesting a password reset

Finally, Figure 15 is showing the new list of posts, which allows the user to filter posts by group, interest topic and tag. In contrast to the old interface, it is now also possible to filter by multiple conditions.

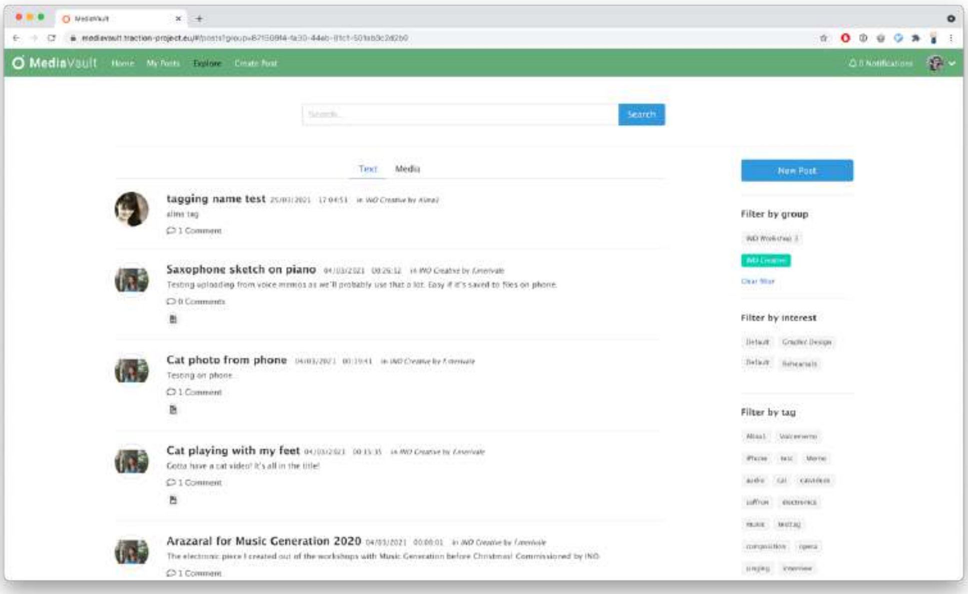


Figure 15: Interface for listing and filtering posts

The production environment of the Co-Creation space is available at https://mediavault.traction-project.eu and the issue tracker, containing all details related to the issues outlined in the previous sections, is available at https://github.com/tv-vicomtech/traction_MediaVault/issues (access provided upon request).

3.2 Co-Creation Stage Features

After the user tests in Leiria and a round of interviews with the SAMP and LICEU teams, the technical partners identified a new set of requirements to develop over the course of the 2nd year of the project. These requirements have been categorised according to their type (UI, Stream related, Universality etc.) and also according to which iteration of the software should provide such functionalities (for the Lisbon show in June, for the 2nd iteration of user tests or for the final release in year 3).

The list of requirements is available in Table 4. For the June 2021 release (prior to the Lisbon show), the developers focused their effort into:

1. improving the UI and UX for the operator;
2. guaranteeing that as many users as possible can access the Co-Creation Stage.

We have been following the same development process described in D2.3 and the one that is used for the development of the Co-Creation Space: all the developers collaborate on Github and they keep track of the requirements, bugs and issues on several Kanban Board organized according to the expected releases of the software. The aim is to complete by June all the issues related to the "Lisbon show" and some of the ones for the 2nd iteration of the user tests. An updated status of the ongoing development process is available on the tool's GitHub repository[2].

Table 4: List of requirements after user tests and interviews with artistic partners.

Type	Description	Priority	Effort
Interface	Add new device type: Single device for camera & screen	High	High
Interface	Add new device type: Only audio	High	High
Universal vs. Professional	Specific interface to control all live cameras and their audios (volume adjustment, etc.)	Medium	High
Universal vs. Professional	Provide a solution to use iOS devices (mobiles and tablets) to be displays or cameras	Medium	Low
Universal vs. Professional	Provide a solution to use Android devices (mobiles and tablets) to be displays or cameras	High	Medium
Universal vs. Professional	Provide a solution to use Windows devices to be displays or cameras	High	Medium
Universal vs. Professional	Provide a solution to use MAC devices to be displays or cameras	Medium	Medium
Universal vs. Professional	Provide a solution to use LINUX devices to be displays or cameras	High	Low
Audio quality	Eliminate the noise-suppression (research how to do it)	Low	Medium
Audio quality	Eliminate other audio filtering such as normalisation, etc. if possible	Low	Medium

[2] https://github.com/tv-vicomtech/traction_RealTimePerformanceEngine (access provided upon request)

Audio quality	Use audio and video separately (as different sources or in the same one but removing sync)	Low	Medium
Interface	Provide an alternative to "NO SIGNAL" in displays (not in the UI of the operator)	High	Low
Interface	Provide specific alternatives to each live stream in the scheduled if there is "no signal"	Low	Low
Stream Adaptation	Combine local / cloud on-demand content: if there is local storage use it, if not use cloud	High	Medium
Stream Adaptation	Improve the usage of bandwidth of the UI of the operator	Medium	High
Stream Adaptation	Adaptive streaming for on-demand content from cloud	Low	Medium
Stream Adaptation	Adaptive streaming for live content	Low	Medium
Stream Adaptation	Lower-latency live audio and video	Low	Medium
Interface	Play/Pause functionalities for all the videos (from the operator side)	Medium	Low
Media	Sync for all on-demand videos	Low	Low
Media	Sync for live cameras - Research if possible	Low	Low
Interface	Timeline definition/synchronisation within a scene	Low	Medium
Interaction	Multiple instances of operators working together	Low	Medium
Interface	Add chat functionality between operator / cameras / displays	Medium	Low
Interaction	Add backline communication (or asynchronous audio message exchange) between participants	Low	Low
Interface	Improve the interface for audio / video capture devices	Medium	High
Interface	Select the video and the audio source to be captured	High	High
Interface	ON-AIR information: show when you are being showed somewhere (main stage, other stage...)	Medium	Low
UI/UX	Improve the initial interface for display devices	Medium	Low
UI/UX	Better management of the scene information: better visualisation on where you are in the scene	Medium	High
UI/UX	Better management of the scene information: better visualisation on how to go to a scene	Medium	Medium
UI/UX	Decide if the components are for each scene, each display or the same for all scenes	Low	Low
UI/UX	Improve the interface for room setup (add devices, etc.)	Low	Medium
UI/UX	Add in the main interface more information about the devices on each location	Low	Low
UI/UX	Add shortcuts for the functionality of the operator	Low	Low

UI/UX	Specify number of scenes and rooms when creating new event	High	Low
UI/UX	Configure the number of scenes rooms in the setup interface of an already created event	High	Low
Media/Interface	Enable streaming of a screen to live streaming service	Medium	Medium
Interface	Add the possibility to control position of videos and live streams in the different layouts	Medium	High
Media	Add the possibility to store on source device the videos	Low	Low
Interface	Add the functionality to open the operator interface in read-only mode	High	Low
Interface	Change the URL	High	Low
Backend	Separate Orkestra server code from Co-Creation Stage backend	High	Medium

3.2.1 User Interfaces

As explained before, the Co-Creation Stage is composed by three applications and interfaces at the client side: The Control, the Viewer, and the Media Provider.

The Control application allows the artists to see all the generated shows (Figure 16A); to create a template for their show and define a number of scenes in a timeline (Figure 16B); to specify the number of locations that will be used as well as devices for each location and their role (viewer and/or media provider) (Figure 16C); and to load pre-recorded media assets (Figure 16D).

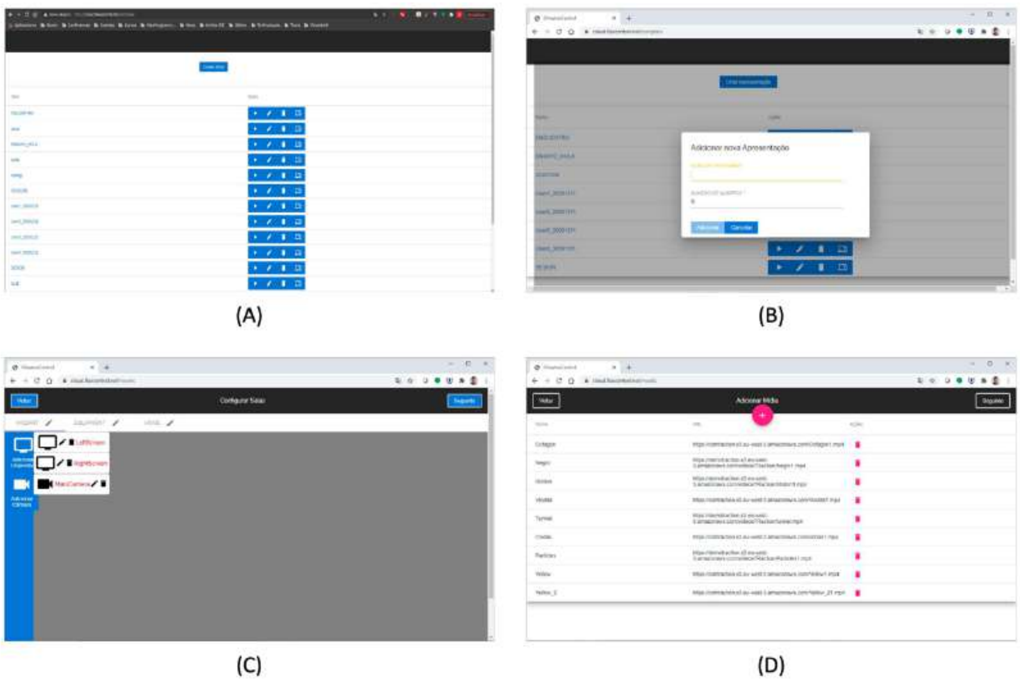


Figure 16: Template generation interfaces for the configuration of an event.

During the show (Figure 17-Theatre, represents the main stage with a musician and two projectors), an operator is able to fully control the Co-Creation Stage and modify on-the-fly all the settings through a specific interface (Figure 18). This interface is divided in five main parts. In the *Media* section, all the available on-demand assets are shown as a preview. There is also a *Cameras* section, where all the defined live cameras preview coming from the Media Providers are shown. The *Preview* section allows the operator to see what is being displayed at each display in each room. At the *Components* section, the operator can assign a specific component to a row (an on-demand video or a real-time stream). Finally, in the *Timeline*, where each column represents a scene, the operator selects the components to be shown (the selected green boxes where the audio can be activated or not) and a User Interface layout is selected for each scene.

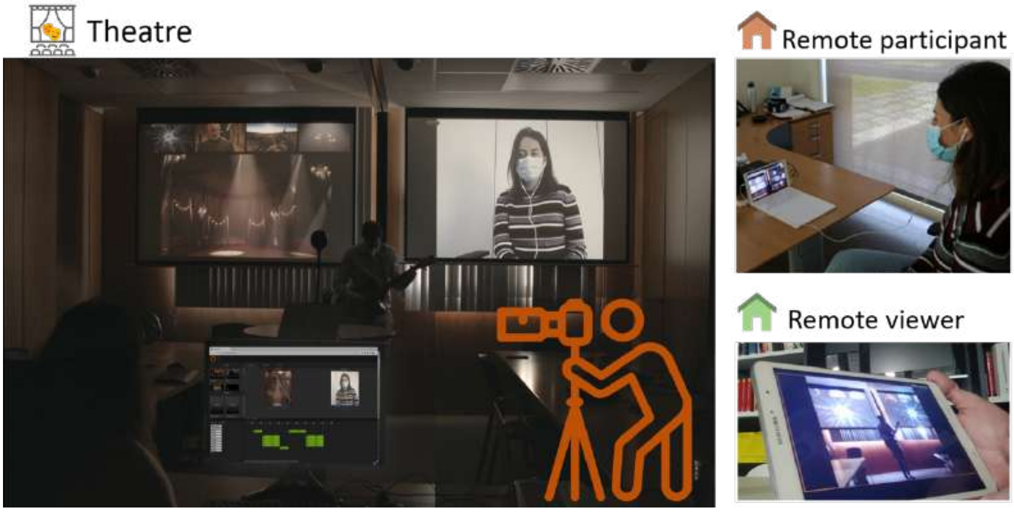


Figure 17: Overview of the Co-Creation Stage in a lab environment.

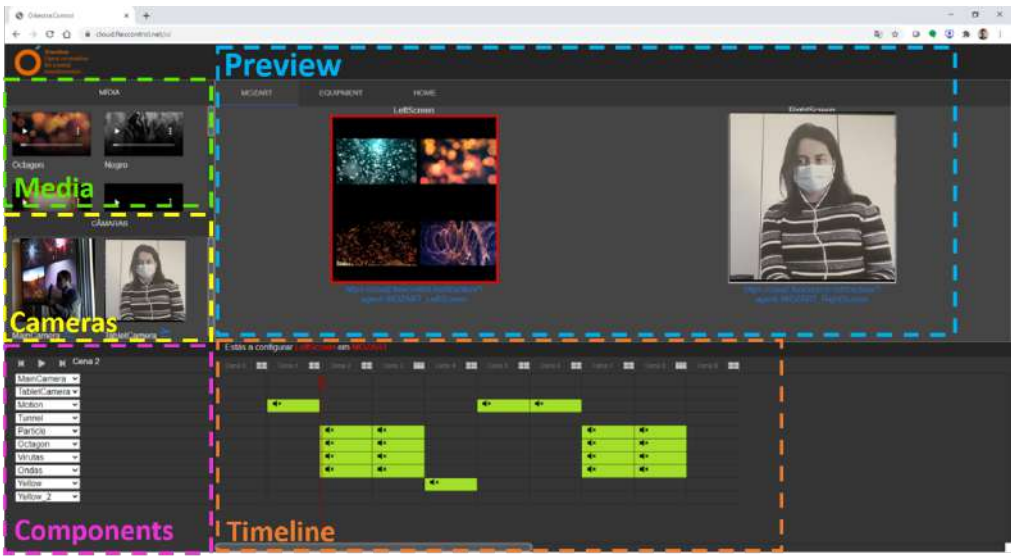
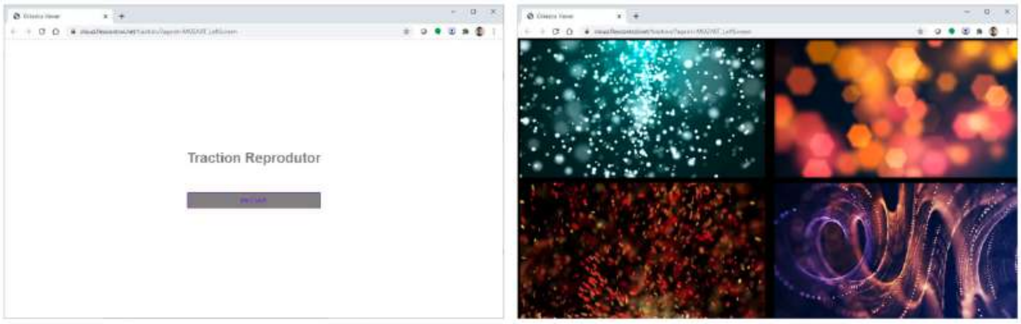


Figure 18: Interface of the Control application for the operator.

In Figure 17-Remote participant, an artist can remotely participate in the show, using a tablet as a media provider and a viewer, following the show in the main stage, and being part of the show across one of the projectors in the theatre. In Figure 17-Remote viewer, a remote viewer can follow the whole show from a tablet. The operator orchestrates all the experience, deciding what to show for the remote viewers, the remote participants and the displays on the stages, moving between scenes and deciding the layout of each output display.

Remote users will follow the show through the Viewer application. When the application is opened in a web browser, an interaction is needed to be able to manage the media of the browser. Once the user presses the start button (Figure 19A), the content is shown in full screen according to the operator decisions (Figure 19B shows a layout with four on-demand media assets).



(A) (B)

Figure 19: Interface of the Viewer application.

Figure 20 shows the interface for the media provider, where the users just need to authorise the use of the camera and the microphone from the Web browser and select the specific device (camera and microphone) they intend to use. From that moment on, the camera is enabled and ready for the operator.

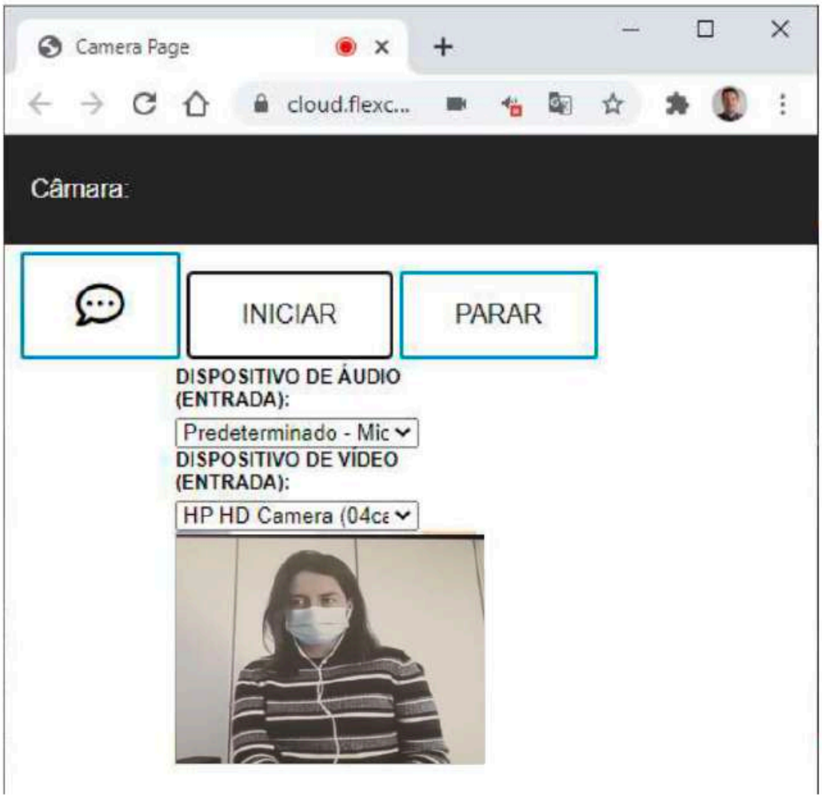


Figure 20: Interface of the Media provider application.

3.2.2 Stream Adaptation

Since network conditions can be highly dynamic, it is very challenging to provide satisfactory user experience during an entire streaming session. Without an effective rate adaption algorithm, users may experience frequent interruptions and significant visual quality degradation. For example, a video with a bitrate higher than the available bandwidth would cause network congestion. On the other hand, a video with a bitrate lower than the available bandwidth might not provide the highest visual quality possible. Therefore, two video delivery adaptation schemes are being designed for the next toolset iteration to help improve the quality of experience of the Co-Creation Stage users. The first approach focuses on the non-real time streams (i.e., videos stored in the Co-Creation Space) while the second approach is used for the live streaming. They both consider different aspects such as network conditions, latency and quality variation.

3.3 Immersive Experience Engine

The Immersive Experience Engine of TRACTION provides solutions for social VR experiences and immersive video viewing. These solutions will be described in the next subsections.

3.3.1 Social VR

As discussed before, the intention of TRACTION is to explore the benefits and drawbacks of Social VR as a novel Opera format. The opportunity is to re-use and extend the work done in the VRTogether project.

The VRTApplication from the VRTogether project is a volumetric video pipeline for photorealistic social VR experiences. It performs real-time capturing, reconstructing, and delivery of volumetric videos of users in the format of point clouds. Users who are at different locations are captured by multiple depth cameras, and their volumetric captures are instantly delivered to the application, allowing distant users to be present in the same virtual space with their photorealistic representations.

Figure 21: A user captured by three Kinect RGB-D cameras

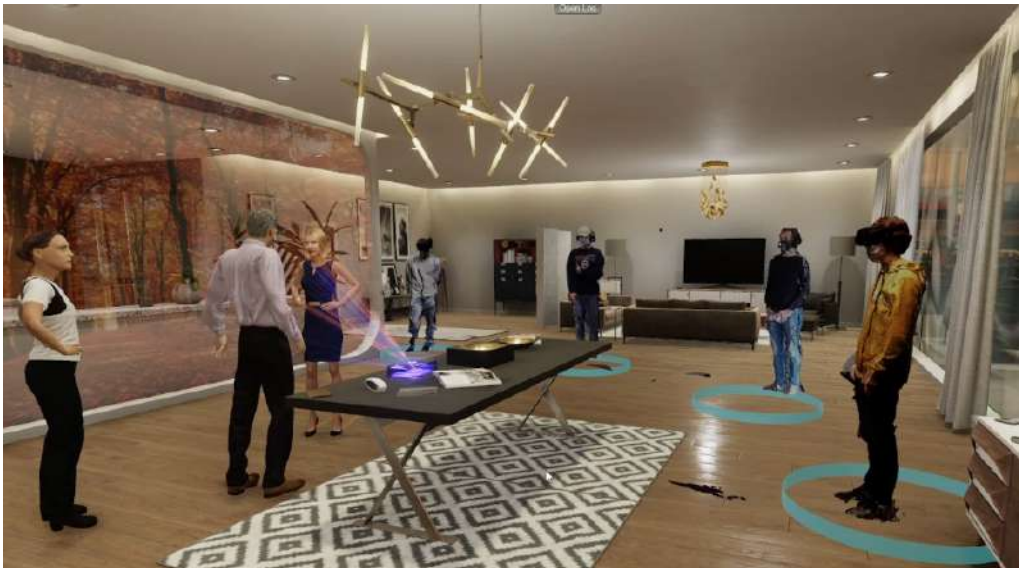


Figure 22: The point clouds of users are delivered to the virtual environment in real time

The volumetric video pipeline allows for low-latency capture and reconstruction of 3D volumes as point clouds, based on a setup of several RGB-D cameras. It also enables the compression and delivery of the point clouds, and the rendering in the receiving side (see Figure 21 and Figure 22). Further details about the infrastructure can be found in the award-winning article “A Pipeline for Multiparty Volumetric Video Conferencing: Transmission of Point Clouds over Low Latency DASH” (Jansen et al., 2020).

VRTogether already explored different use cases for their application: watching together an interactive immersive movie, blow out the candles on a birthday cake, or a medical examination room where the patient joined from a park, using her smartphone and the 5G network[3]. The use case we are considering would be to use the VRTApplication so users can meet and interact to each other in the lobby previous to the virtual Opera, and then move to the main hall for watching the novel type of Opera.

3.3.2 Accessible 360° Player

Currently, the immersive video player of TRACTION can execute regular 2D and 360° content. Users can navigate in immersive videos by dragging the point of view with a mouse, touch screens or head movements in virtual reality headsets. Figure 23 presents a screenshot of the video selection list of the player. Figure 24 presents the footage of a 360° video being played.

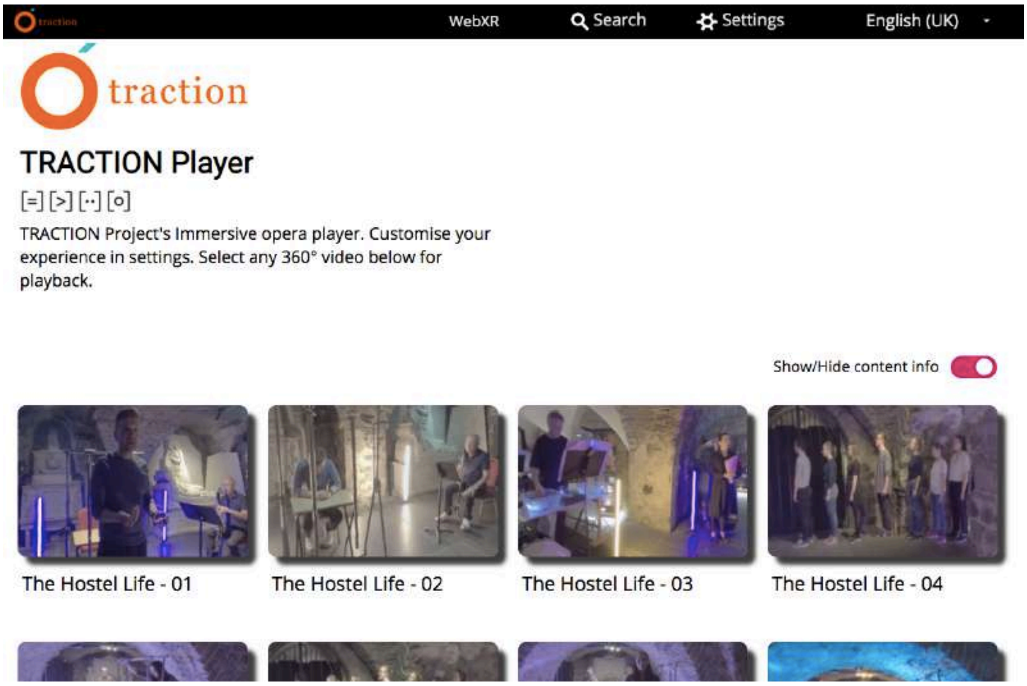


Figure 23: Accessible 360° Player video selection list.

The player contains a number of tools for accessibility, including subtitling, (spatial) audio description, sign language interpreting, voice control and large menus for visually impaired people. These can be accessed through the main player menu as shown in Figure 24. The

[3] https://vrtogether.eu/2020/11/09/worlds-first-volumetric-video-conference-point-clouds-over-a-public-5g-network/

four icons [=] [>] [··] [o] represent text subtitles, sign language, audio subtitles and audio description, respectively.

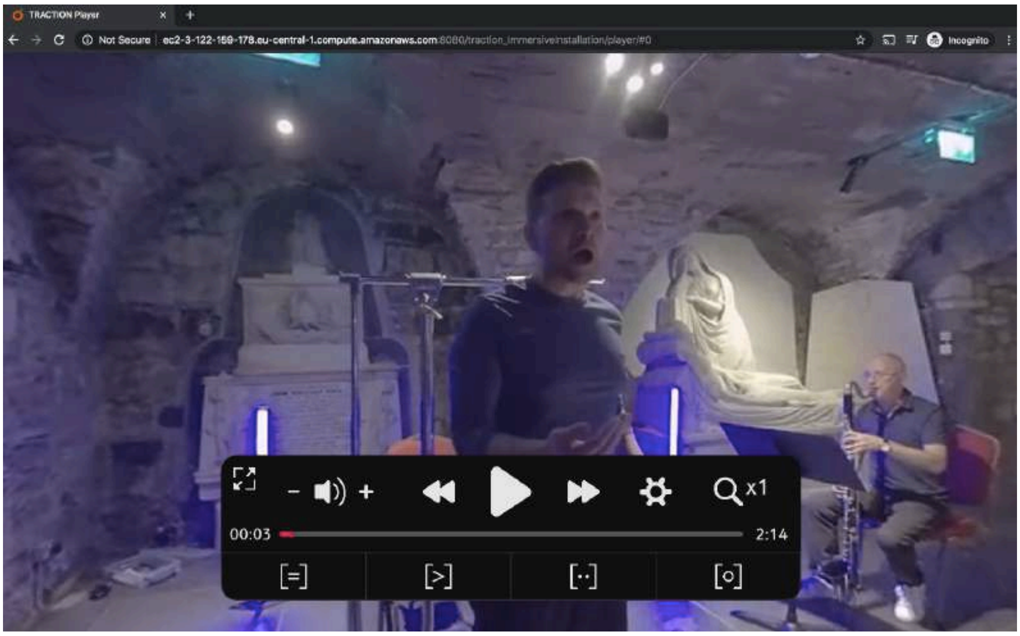


Figure 24: An example of a video playing on the Accessible 360° Player.

In the context of the TRACTION project, the player can be used for the dissemination of opera content in a wider range of devices, such as smartphones, tablets, laptops and desktops, since it is a browser-based immersive media player.

4 Architecture of the Toolset

This section focuses on the new architectural changes to support the functionalities described in section 3 and future functionalities.

4.1 Co-Creation Space Features

The following sections highlight updates in the architecture of the Co-Creation Space since the last iteration. Particular focus will be given to updates in the media playback functionality of the front-end, updates of the back-end data model, facilities for tracking and analysing user behaviour related to the open trials and updates to the deployment environment.

4.1.1 Improved Video Streaming

During development of the feature for adding multiple media items to a single post, an issue with video.js, the library that was used to enable adaptive streaming, was discovered. Through research it was discovered that this specific bug had already been logged with the developers of video.js a significant amount of time ago, yet no solution had been provided as of yet. Unable to work around this issue, the decision was taken, to replace video.js with a much more low-level library, namely DASH.js. In contrast to video.js, which supplies the developer with a fully-fledged player, DASH.js merely manages the streaming of the desired content and contains no actual media player itself. This meant that we had to implement our own player, controls and facilities for rendering subtitles.

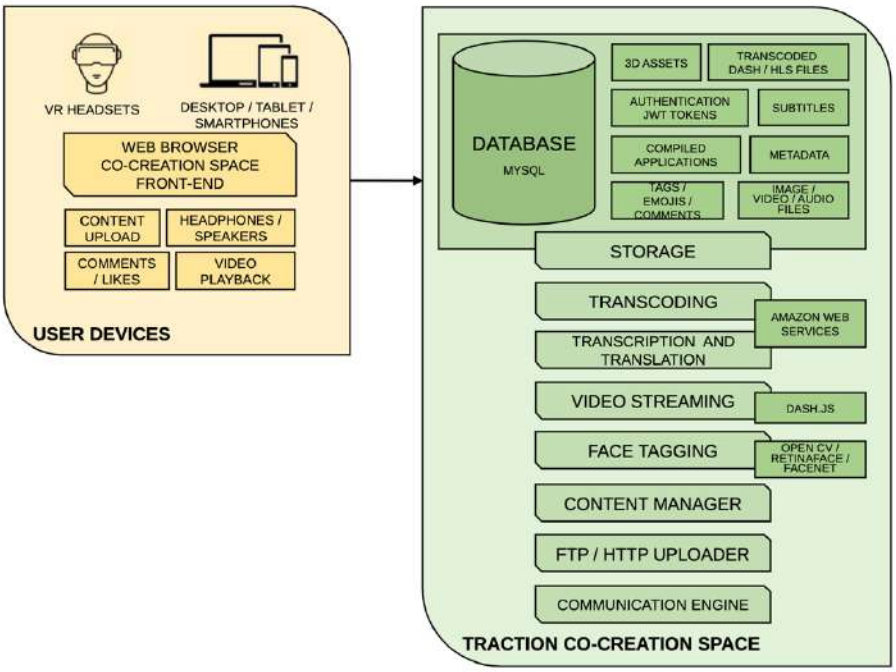


Figure 25: General architecture of the Co-Creation Space

Moreover, it was discovered that iOS mobile devices (expect tablets) have no support for DASH, but do support HLS natively. Because of this, uploaded media items are now encoded in both, DASH and HLS and the application performs feature detection when a media item is loaded and either starts a DASH stream if supports and otherwise defaults to native HLS streaming.

Though all of this incurred some amount of unforeseen implementation work, we now have a much leaner, more flexible playback pipeline, which is more performant across a greater variety of platforms. It gives us the ability to implement additional features in a simpler way and significantly reduces the amount of library code that has to be transferred to the client.

4.1.2 Updated Database Model

During the development process, there have been multiple changes in the relational data model of the application, with entities being added, deleted and modified.

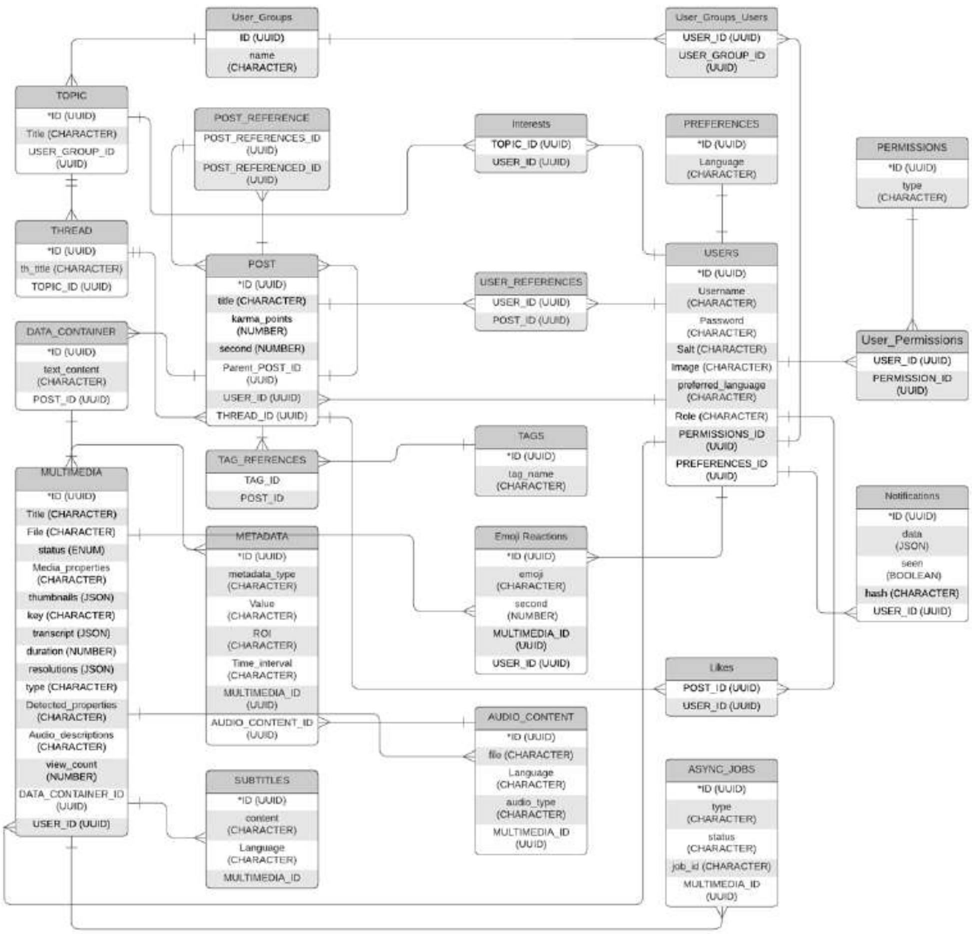


Figure 26: Updated ER-diagram of the database

Figure 26 illustrates the current database model of the Co-Creation Space. Though some properties on some tables have changed since the last report, the models of the tables themselves largely remain then same. There are, however, two new tables that were introduced in this iteration and are worthy of discussion. First, an entity named `Async_Jobs` has been introduced, which forms a one-to-many relationship with the entity `Multimedia`. Its purpose is to represent asynchronous processing jobs on multimedia records and their status, e.g. DASH and HLS transcoding or video/audio transcription jobs. This new entity replaces the now obsolete properties `transcription_job_id` and `transcoding_job_id` on the entity `Multimedia` and allows us to have a variable number and type of asynchronous processing jobs per multimedia record and independently track each job's status. While this is useful from a back-end perspective, it does not affect the end user in any meaningful way.

The other change worthy of note is the introduction of the entity `User_groups`, which allows the creation of groups and users. It affects the end user in a significant way. The entities `User_groups` and `Users` have a many-to-many relationship, meaning that a user can be a member of multiple user groups. Another part of the application that is affected by this user group feature are interest topics. Instead of being global, each interest topic is now associated to a specific user group and by extension, this implies that also each thread, post and multimedia item is implicitly scoped to a user group. This allows us to create multiple sub-communities, which are isolated from each other, but can be managed by a select group of users which are members of multiple communities at the same time.

4.1.3 User Behaviour Tracking & Analysis

For the open trials it was of great importance to learn as much about how users used the Co-Creation Space as possible. This meant the integration of a variety of features that allow the tracking of user behaviour while using the application. This ranges from, client-side aggregate metrics to more specific metrics that track how users interact with media items, how they navigate the pages and how they interact with other users.

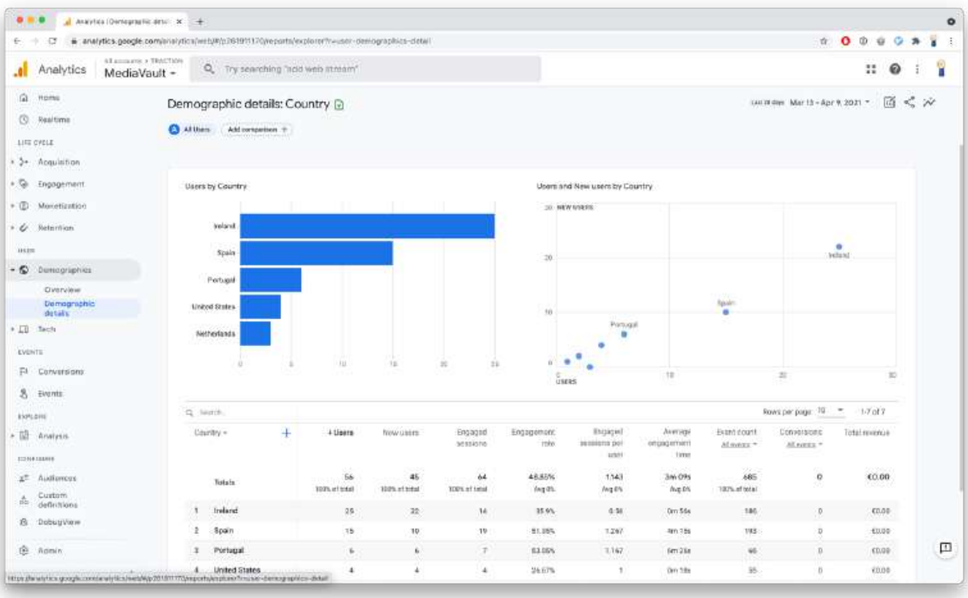


Figure 27: Google Analytics tracking aggregate global metrics of user behaviour

Broadly speaking, the metrics for tracking user behaviour can be divided into two categories: global and user specific. Global aggregate metrics are measurements collected anonymously through the browser by a third party. This type of service is provided through Google Analytics. Figure 27 shows an example of metrics related to demographics and geo-location collected through Google Analytics.

Another type of metrics that we collect are metrics tied to a specific user. These are collected internally in the application and provide insight into how users interact with the application itself, media items and with each other. These types of metrics are collected by custom application code in the front-end and stored in the application's back-end database. Table 5 shows an overview of the types of metrics that are collected.

Table 5: Internal metrics that the application collects

Type	Description
View count	A numeric count of how many times a user views a certain media item. For video and audio, a view is counted if the user interacts with the media item for a minimum total of 30 seconds.
Media interactions	Interactions with media items. This includes playback start/pause, seeking, activating/deactivating subtitles and switching subtitle tracks.
Search queries	Types of keywords used when interacting with the internal search engine, as well as number of results generated by the search query
Internal navigation steps	Navigation path that a user takes through the application. This includes sequence of visited pages, forward/backward navigation and filter parameters applied to lists.
Computed metrics	Metrics computed from data in the database. This includes data such as posts created, comments written, media uploaded and interactions with other users.

Computed metrics are generated on demand and aggregated from data in the database using SQL queries. All other metrics listed in the table, however, are either their own entities, or form part of other entities and can be accessed directly. Figure 28 shows these entities, their properties and how they are related to the `Users` entity in the database. For conciseness, all unrelated properties are omitted from the `Users` and `Multimedia` entities. `Internal_navigation_steps` stores each navigation step on the page taken by the associated user in its `Data` field. Additionally, it stores the current user-agent, enabling us to deduce the type of device that was used at that point in time. `Search_queries` stores information about search queries carried out by a user and the number of results obtained by the site's search engine. `Multimedia_interactions` stores information about how a certain user interacts with a certain multimedia item (i.e. a video) in its field `Interaction`. This includes playback actions, activation of subtitles and changing to full-screen mode. Finally, the entity `Multimedia` stores a view count for each record as a numeric property.

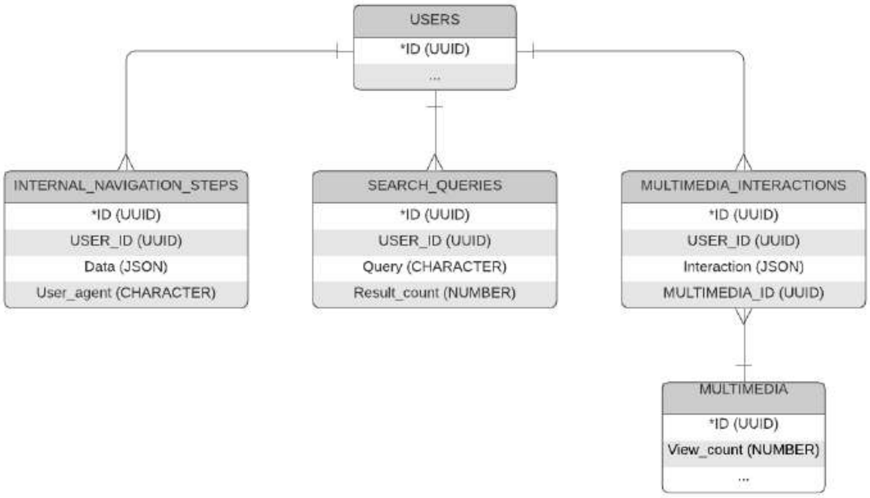


Figure 28: Entities for collection of user behaviour data and their relation to `Users` entity

Figure 29 shows how each of the metrics is collected, stored and retrieved. Data for Google Analytics is collected by the Google Analytics JavaScript plugin and sent to Google Analytics server. Internal metrics are collected through JavaScript code in the application itself. So, for instance media interactions are collected by wrapper code around the media player, which listens for playback event carried out by the user. After such an event is triggered, a HTTP request with the relevant is sent to the corresponding endpoint on the server, which then stores the entry in the database and associates it to the user that carried out such action.

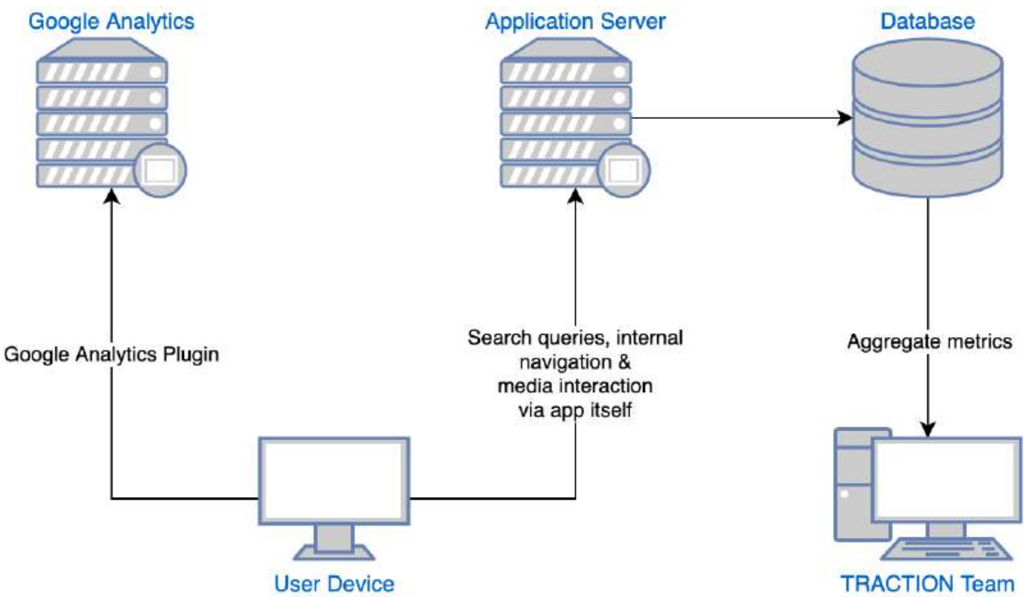


Figure 29: Workflow of user data collection and retrieval

Similarly, internal navigation is tracked by code that is wrapped around the application-internal router. This way, every time a navigation step on the page is taken by the user, it is intercepted and a record of it is forwarded to the server and stored in the database before the step is actually carried out.

Records about search queries are stored by the endpoint responsible for carrying out searches itself, so no additional HTTP interaction is necessary. This way, it is also possible to store the number of results with the record.

Whereas data gathered by Google Analytics is accessed through the Google Analytics website, internal metrics are retrieved directly from the database by querying the relevant tables through the use of SQL, or the export of such data to CSV, or a similar table-based format and its processing with the help of the programming languages Python or R.

4.1.4 Production Environment

In order to better facilitate the open trials, the decision was taken to create a separate deployment environment in parallel to the development environment. This new production environment was to run on the same physical machine as the existing development environment, interact with the cloud environment in the same way and be served by the same reverse proxy which controls the entry point open to the public internet. It would however have a private copy of the database and naturally be reachable from a different domain name.

Different from the development environment, which is continually redeployed from the head revision of the develop branch of the git repository, the production environment would only be deployed from tagged revisions of the master branch. This has the advantage that it allows for an additional level of quality control as only previously reviewed and stable versions are deployed to the environment. Managing two different environments incurred additional complexity and necessitated the employment of a more sophisticated container orchestration system.

Figure 30 shows the updated deployment setup, employing Docker Swarm as container

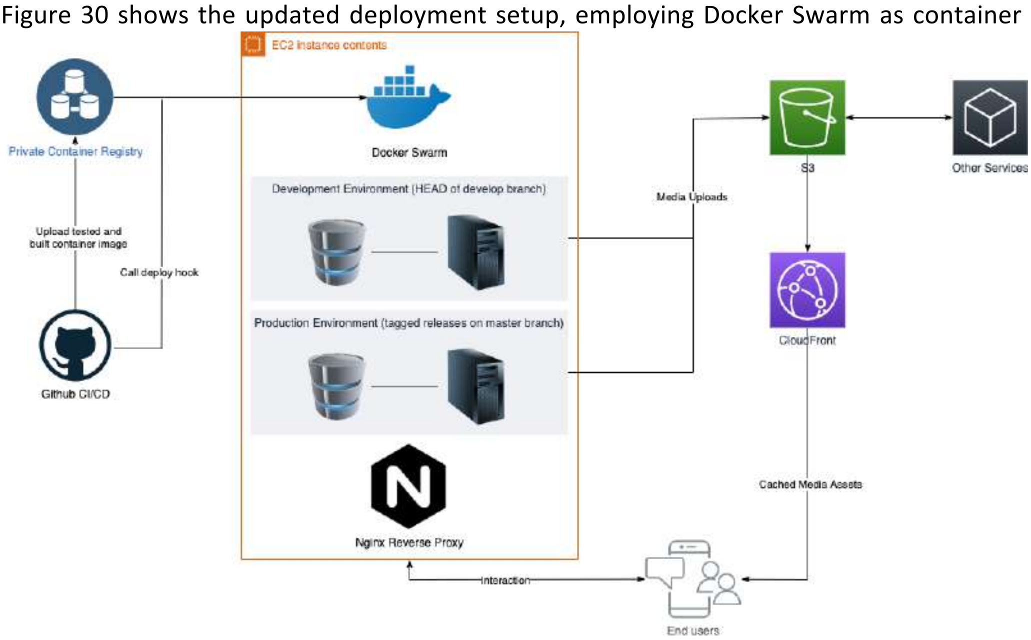


Figure 30: Setup of the production environment

orchestration system. If new revisions are pushed to either the head of the develop branch or a new version is tagged on the master branch, Github's CI/CD system, builds a new container image, lints the code, runs the test suite and if everything passes, uploads the newly built container image to the private container registry. It the invokes the corresponding deploy hook on the container orchestration system via a HTTP POST request. The container orchestration system then downloads the latest image or image with the given version from the private container registry and initiates the changeover from the image that is currently running. Once the new image is spun up, both, the new image and the old one run in parallel for 10 seconds to ensure a seamless changeover until the old image is terminated and disposed of.

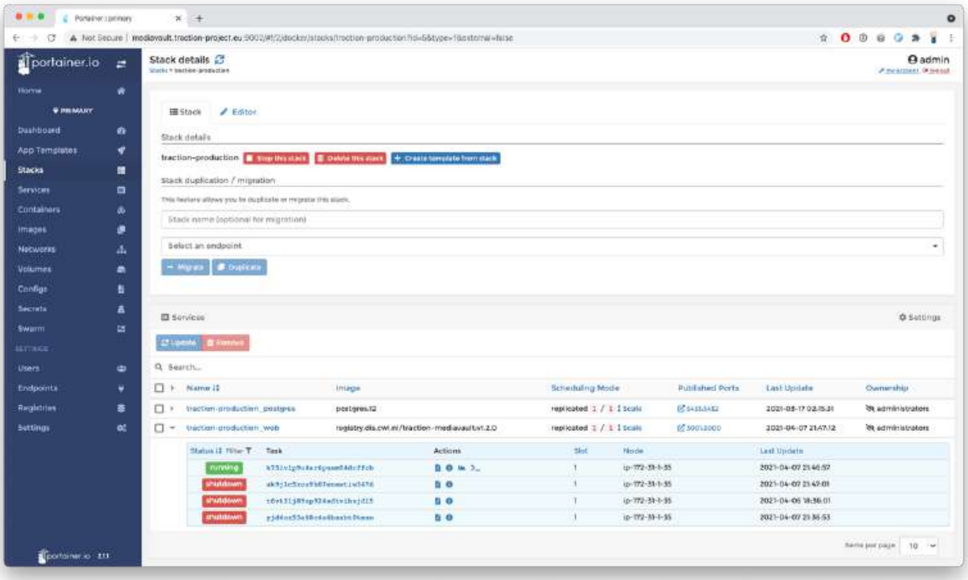


Figure 31: Web interface of the container orchestration system

Docker Swarm also comes with a web interface which allows for the managing of each environment, virtual volume, container and image, making debugging easier. It also simplifies management of application secrets, network and service configuration management. Figure 31 shows the web interface of Docker Swarm, listing the current status of the production stack. The container orchestration system also makes replication of distinct services as well as scaling of entire stacks across an arbitrary number of physical compute nodes more straightforward, should the need for it arise, for instance due to an unforeseen spike in traffic.

For accessing two environments now, a new domain name had to be registered. Whereas previously, when only the development environment existed, it was available under https://mediavault.traction-project.eu, now the development environment is pointed to by https://mediavault-dev.traction-project.eu and the production environment responds to https://mediavault.traction-project.eu and as previously, unencrypted requests are automatically redirected to a secure connection.

4.2 Co-Creation Stage Features

The Co-Creation Stage is based on a client-server architecture on top of Web technologies and uses a platform that allows the orchestration, communication and distribution of content between different devices and multi-device interaction.

The Co-Creation Stage is therefore made of two components: the client side composed of three applications (Control, Viewer and Media Provider) and a server-side implementation. The client side contains most of the application logic, distributed in its three applications:

- The *Control* application provides the interface for the operator allowing to create the event, to manage the sources (cameras and videos) that it will contain, as well as the devices that will show the content. It is also able to customise the layout template that each device will show and act on them in real-time
- The *Viewer* is a simpler application for visualising the content on each display according to the orchestration decided by the operator.
- The *Media Provider* offers the functionalities required to integrate the live audio and/or video streams.

The *Control* application is based on a user interface that allows the operator to manage the rooms (stages, locations, etc.) and their devices (displays or projectors on a stage, a mobile device, etc.), the role of each device (viewer and/or media providers), and the multimedia assets. The operator can create a timeline structure that defines which components are associated to which sources and how they will be represented in each device on each location. This information is shared in real time with each connected device through a Shared Context. Furthermore, there is a timeline timestamp to synchronise all the content which is propagated using Timing Server.

The *Viewer* is built on top of media components. According to the orchestration defined by the Control, a viewer visualises *N* media components, creating a specific layout assigning size and location to each of the components. Each media component is able to render real-time streams through WebRTC, on-demand videos (such as mp4 files, HLS or MPEG-DASH), images, audios, etc.

The server side, on the other hand, is composed by several modules that provides functionalities accessed by all the clients:

- A Rest API for the manipulation of data persistence;
- A Websocket Server, for sharing the context of the application in real-time;
- A Timing Server for sharing a timer which is synchronized to millisecond accuracy;
- A WebRTC gateway that allows publishing and consuming real-time media flows.

The chart in Figure 32 shows the Co-Creation Stage architecture, its client-server structure and its relationship with additional server providing storage and real-time communication capabilities.

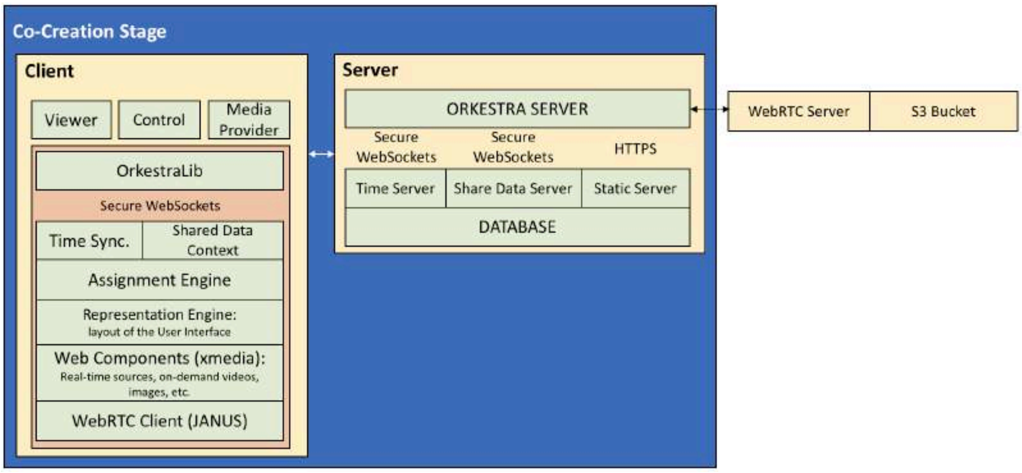


Figure 32: Co-Creation Stage architecture

4.2.1 Stream adaptation

There are two types of video and audio streams supported by the Co-Creation Stage: Non-Real Time and Real-Time streams. Each type of stream requires a specific solution for content adaptation, which is necessary for devices with different specifications and network capabilities.

4.2.1.1 Non-Real Time Stream Adaptation

The non-real time streaming adaptation scheme is based on the MPEG-DASH standard. It is deployed at the Web Component entity (see Figure 32) to make decisions on the bitrate to be used for pre-recorded streams. This information is then shared with the rest of the users through the Shared Data Context. To this end, the non-real time streaming adaptation scheme has 5 modules, defined as follows:

- **Device Monitor:** collects the user device specifications from the Shared Data Context. This includes the device type and the supported resolution.
- **Bandwidth Predictor:** forecasts the available bandwidth in the near future based on the information collected from the network.
- **Buffer Monitor:** keeps track of the occupancy of the buffers storing the downloaded segments.
- **Quality Variation Monitor:** stores the difference in bitrates among the previously downloaded segments.
- **Adaptation Unit:** selects the bitrate of the next segment to be requested based on the information received from the aforementioned modules.

Figure 33 describes our DASH-based adaptation solution for non-real time streams. The MPEG-DASH client requests segments for the different videos using HTTP GET requests. The segments are forwarded by the MPEG-DASH server, deployed in the Co-Creation Space, through the network. The communication manager can be any entity that allows communication between the client and server, such as a reverse proxy. The downloaded segments are stored in the playback buffers. Before requesting a segment, the Bandwidth Estimator computes the network available bandwidth and shares it with the Adaptive Unit.

This latter uses this information along with the buffer occupancy, provided by the Buffer Monitor, the bitrates used for the previously downloaded segments, stored in the Quality Variation Monitor, and the device specifications, provided by the Device Monitor, in order to select the bitrate of the next segment to be requested from the server. When selected, the Adaptive Unit informs the Quality Variation Monitor and the Shared Data Context. This latter informs the other users about the recently selected bitrate. Finally, the segment request is sent to the DASH server to start the forwarding process.

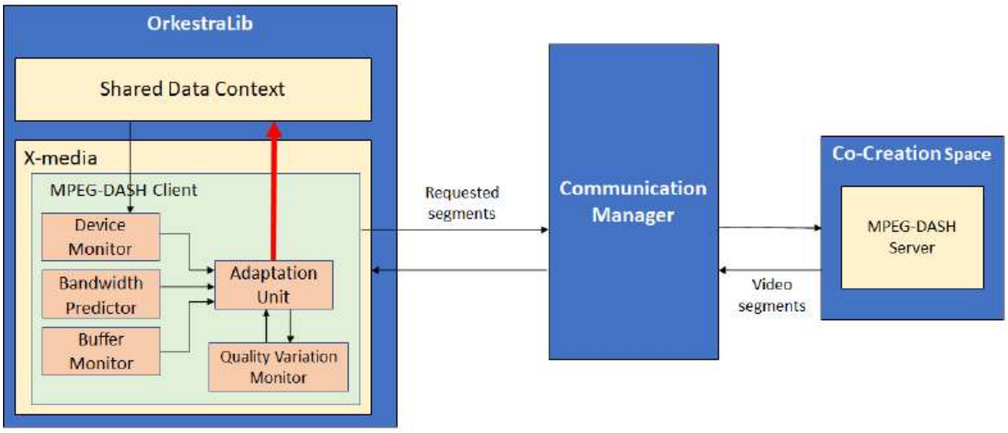


Figure 33: Architecture of the non-real time streaming adaptation solution

4.2.1.2 Real-Time Adaptation

The real-time streaming adaptation scheme is also deployed at the Web Component (X-Media) entity of the Co-Creation Stage. The scheme aims to adapt the bitrate of real-time sources from live camera, which are transmitted with the WebRTC specification, which enables real time media communications between browsers and devices. These live videos are given to the WebRTC client for streaming.

WebRTC Internal Statistics related to send/receive bitrate, available bandwidth, delay, packet loss are collected by the WebRTC client via *RTCPeerConnection* and *getUserMedia* APIs. Figure 34 illustrates the adaptation solution for real-time streams.

The Real Time Adaptation Engine in the *Device Monitor* requests information from the Shared Data Context in relation to target *device type*, such as computer, tablet, mobile and projector; *resolution*, indicating the highest video bitrate supported; *audio* bitrate and type (i.e. mono, stereo) and the highest audio bitrate supported. *Layout* information such as the number of windows in layout and the size of each window is also used. The Real Time Adaptation Engine also estimates the quality of real-time video for video width, height, frame rate through the Real-Time Video Quality Estimator.

The WebRTC Session Monitor requests WebRTC session stats for agreed frame rate, codecs, bit rates, and network conditions. This information is fed into the Network Condition Estimator / QoS Predictor, which predicts the network bandwidth, delay, send/receive bitrate of the streams.

The WebRTC Internal Statistics are used to avoid recalculation of the bandwidth as it may delay the streaming, while the Adaptation Unit decides the bitrate based on the Network Condition Estimator and the Real-Time Video Quality Estimator for a given session.

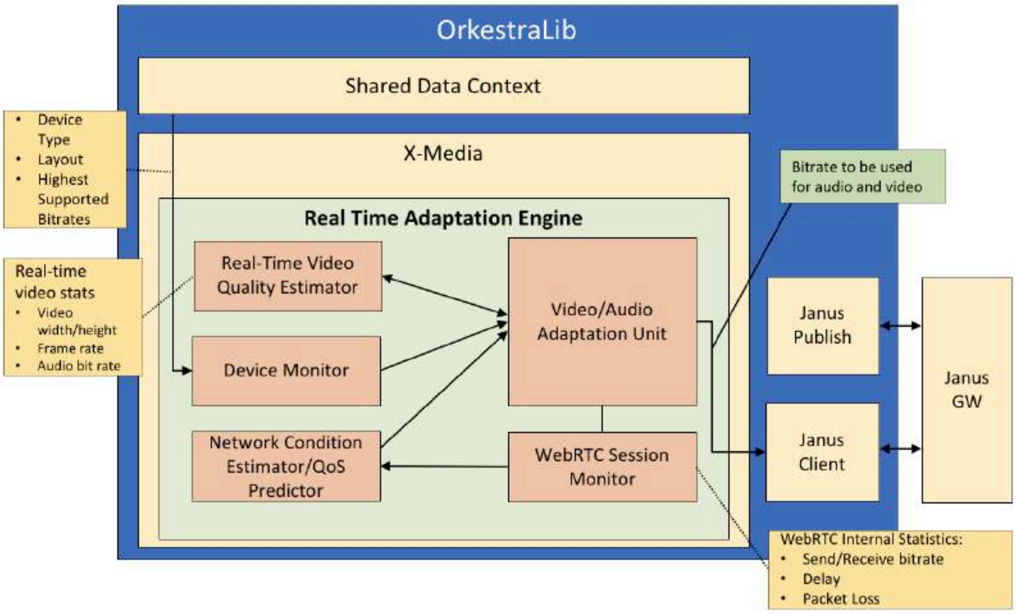


Figure 34: Architecture of the real time streaming adaptation solution

4.3 Immersive Experience Engine

The two components of the Immersive Engine Experience of TRACTION require distinct architectures, which are presented in this section.

4.3.1 Social VR

The VRTApplication is based on Unity. The architecture of the pipeline has been designed and implemented in a way to allow easy replacement of modules, in order to experiment with different capturers, encoders and transport pipelines. It also allows for adaptation to different use cases and budgets.

In particular we are interested in the point cloud pipeline (Figure 35). For each instance of our architecture there is a single transmission section (tile, compress, sender), and an independent reception section per other participant (receive, decompress).

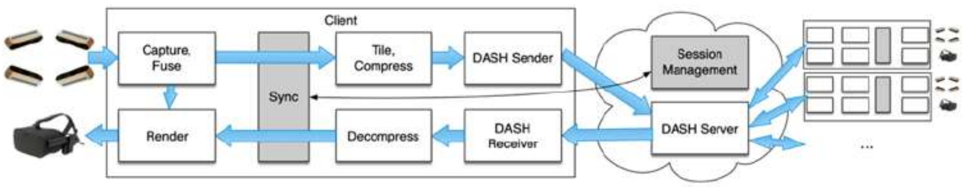


Figure 35: The architecture of the Point Cloud Transmission Pipeline

4.3.1.1 Capture and fusion

The capture module interfaces to the cameras and captures RGB and Depth images, which are transformed using the intrinsic and extrinsic matrices and converted to point cloud representations. These point clouds are fed to the tiling and compression module. Tiling splits a single point cloud into a small number (1-8) of non-overlapping point clouds that together comprise the whole original point cloud. Through tiling we enable viewers to save bandwidth by only downloading the relevant parts of a point cloud, for example omitting invisible tiles. Subsequently each tile is downsampled into a small number (1-3) of resolutions. This downsampling is generally done through voxelization: overlaying a 3D cubic grid over the point cloud space and replacing all points within each cube by a single voxel with the average position and colour.

4.3.1.2 Transmission

The resulting point clouds are fed into a lossy compressor which creates a linearized compressed data block for each tile at each resolution. The DASH sender is instantiated with a description of the number of tiles and resolutions and creates a manifest file based on this information, which is uploaded to the DASH server. Subsequently, as compressed data blocks become available, they are uploaded as DASH segments and the manifest file is periodically updated. The DASH server stores the manifest file and the segments and serves these to the receivers over HTTP or HTTPS.

4.3.1.3 Reception

The DASH receiver downloads the manifest file and makes the information on available tiles and compression levels available to the renderer. Based on view point, gaze direction and distance between viewer and subject the renderer selects the tiles it wants to receive and the quality level for each of those, and the DASH receiver starts requesting the segments for these tiles in parallel.

The per-tile compressed data blocks are fed to the decompressor which converts them back to point clouds. The synchronization module is responsible for synchronizing the tile streams and the audio.

4.3.1.4 Renderer

Finally, the renderer receives the self-view point cloud from the capturer and a number of tiles per other participant and renders these in a 3D space based on the viewpoint and position of the participants.

4.3.2 Accessible 360° Player

The Accessible 360° Player of TRACTION provides a user-friendly interface and web-based tools that work across multiple devices, so users can experience content on browsers of mobile phones, desktops, VR headsets and virtual reality domes.

Figure 36 illustrates the architecture of the player. A web server hosts the accessible 360° player and videos and metadata are hosted in the Co-Creation Space database and its AWS container. Other important features of the immersive environment include adapting the content for different devices, network capabilities and screen sizes, and present accessible features to users such as subtitles.

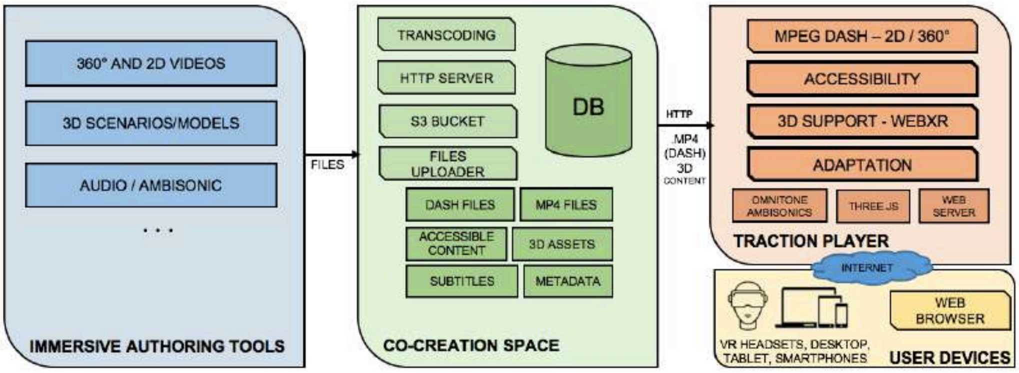


Figure 36: Accessible 360° Player architecture

The player employs the MPEG-DASH standard. This standard allows the playback of 360° videos at different quality levels according to network conditions and device type. One of the possibilities in relation to adaptation in the player is to reduce quality in areas that the viewer is not likely to look at, while the main areas of the 360° video, such as the location of the singer, will suffer less or no degradation when network conditions are constrained.

Other approaches for adaptation include bitrate streaming adjustments, by controlling the buffer occupancy, available bandwidth, and predicting network variations. It is also important to give priority to the audio quality in the adaptation process, as the player focuses on opera content.

5 Pilot Activities and Requirements Gathering – Year 2

This section details the plans for the next pilot activities, focus groups and evaluation sessions for requirements gathering.

5.1 Co-Creation Space

In the second year of the project, we plan to gather requirements for the Co-Creation Space through an open pilot of the tool, paired with a series of co-design sessions with designers followed by focus groups with trials.

Open Pilot:

Over a two-month open pilot, we will collect requirements with creative teams and workshop participants from the three trials. During the open pilot, creative team participants will use the co-creation space to share and discuss media content in a free form. In complement, workshop participants will use the co-creation space to complete various co-creation processes. Data from the open pilot will be used to understand individual and group behavior patterns and will inform requirements about the usability of the tool.

Codesign Sessions:

The codesign sessions will inform the design of the next iteration of features. There will be four codesign sessions. The first session will consider smart upload features that make it easier for Co-Creation Space users to upload and share different kinds of media. This may include automatic suggestions for trimming photos, touching up colors, and preview features. The second session will consider media commenting features, such as highlighting, underlining, or drawing on media.

In the third co-design session, we will consider what sort of privacy and permissions can help support the goals of sharing content, in a way that makes all members of the community---the trial leaders, members of the creative team, and participants from different groups---feel comfortable posting, reacting, and sharing content. Finally, in the fourth co-design session, we will consider how to design a “narrative engine” to automatically summarize the codesign data in a meaningful way, to support both participants' progress and external evaluation.

Each session will take approximately 2 hours, and will include a codesign task, followed by a requirements discussion. After gathering requirements from the individual codesign sessions, we will conduct focus groups with trial leaders to gather an additional round of feedback and requirements.

5.2 Co-Creation Stage

For the second year of the project, the Co-Creation Stage will be tested in two capsule projects developed by SAMP. The first will be a two-day performance in Lisbon (at the Gulbenkian Foundation theatre) and the second a two-day performance in the Leiria prison. The performance in Lisbon is especially relevant since it will feature two co-located stage (the one in the theatre an another one in the prison) and three remote users who will connect from Lisbon, Dublin and Brazil.

During the performance the tool will be tested for latency and bandwidth usage, while after the test we will conduct short interviews with the operator of the Co-Creation Stage in order to check how useful were the new features and also to gather feedback about the user experience as well as the technical capabilities of the tool. We also plan to ask few selected users (among the audience and the remote participants) to fill a short questionnaire about the way the Co-Creation Stage enabled the feeling of connection between the participants.

The data gathered in Lisbon will then be analysed in order to extract new requirements for development in the second half of the year. Once the development is complete, we plan to run a new set of user tests to further evaluate the UI and UX of the tool.

As the performance in Leiria will be a simplified version of the one in Lisbon (there will be only one main stage, and the remote participants will be the same), the current plan is to evaluate and collect user feedback only in Lisbon.

The data gathered in Lisbon will be very valuable for LICEU, which will perform a showcase of the Co-Creation Stage in 2022.

5.3 Immersive Experience Engine

The plans of the second year of TRACTION in relation to the Immersive Experience Engine components, the Social VR and the Accessible 360° Player, include development, enhancement and testing of the tools.

5.3.1 Social VR

In the first year, we ran three open-ended focus groups in partnership with INO trial leaders, and experts in opera and VR. These focus groups helped us to consider how creative, sensory, and social VR technologies may be used to develop a new form of immersive opera. The qualitative findings suggested the importance of 4 themes: 1) creating an immersive and transformative experience, 2) giving audiences a sense of agency over their decisions, 3) democratizing the experience through onboarding and accessibility considerations, and 4) considering the virtual representations of the performer and audience bodies, the sense of ceremony and occasion, and the social space itself. These findings informed a set of requirements for the immersive opera experience, and for the related social rituals.

In the second year of the project, we plan to build a SocialVR experience that makes these social rituals a part of the immersive experience, by building a VR model of a lobby space. We will create an immersive social VR opera experience, where the audience will be represented realistically, using volumetric video. After prototyping these virtual representations, we will test them in a lab study to understand the value of the immersive experience, participant representations, and interactions within the social VR space.

5.3.2 Accessible 360° Player

The plans for the second of year of the project in relation to the accessible player focus on the development of new features to support the dissemination of immersive opera content.

The player interface will be extended to support more languages such as Portuguese and Irish, as English, Spanish, German and Catalan are already supported. Other development tasks will be related to approaches for improving the experience while using the player, in relation to video and audio quality. 360° videos normally have large file sizes, therefore, the player needs to employ approaches that reduce buffering and loading times in devices in congested or limited networks.

Studies can also be performed to evaluate the benefits of the player in terms of usability and user perception regarding immersive opera content. These studies will help us gather requirements for the next iteration of the tool and understand which features need more attention.

6 Conclusion

This deliverable presented the second iteration of the toolset of the TRACTION project, outlining the intermediate features, requirements and updated architectures of the three main components of the TRACTION toolset: The Co-Creation Space, the Co-Creation Stage and the Immersive Experience Engine, containing the Social VR solution and the Accessible 360° Player.

The intermediate toolset implementation contains the new features included after the pilot activities of the first year of the project. These features consist of improvements correcting issues detected on feedback from users and new functionalities based on requirements gathered during focus groups and open pilots.

The next steps for the toolset development include further testing via pilot activities, feedback gathering and the extraction of new requirements for development in the second half of 2021.

Deliverable D2.5, which is scheduled to be finalised in November 2021, will consolidate the technical requirements identified during the project's second year and present the plan for the final version of the toolset to be completed in the third year of the project.

The next iteration of this deliverable (D2.6) is going to present the final prototype of the toolset and is planned for June 2022.