Grant Agreement No.: 101016509
Research and Innovation action
Call Topic: ICT-40-2020: Cloud Computing

CHARITY
Cloud for Holography and Augmented RealITY

Cloud for Holography and Cross Reality

D1.2: Reference scenarios, use cases and requirements

Version: v1.0

Deliverable type	Report
Dissemination level	PU (Public)
Due date	30/06/2021
Submission date	06/07/2021
Lead editor	Antonis Protopsaltis (ORAMA)
Authors	Joao Rodrigues (DOTES), Alex Roibu (HOLO3D), Antonis Protopsaltis (ORAMA), Manos Kamarianakis (ORAMA), Zbyszek Ledwon (ORBK), Enrico Zschau (SRT), Thu Le Pham (UTRC), Michael McElligott (UTRC)
Reviewers	Patrizio Dazzi (CNR), Lorenzo Blasi (HPE), Tarik Taleb (ICTFI)
Work package, Task	WP1, T1.1
Keywords	Reference scenarios, use cases, requirements

Abstract

This document describes the reference scenarios, use cases and their requirements in CHARITY project. For each of the use cases that will be used to test the CHARITY framework, the current as well as the envisioned edge-based architecture is provided. Reference scenarios are described, and a list of potential general purpose, and use case functional/non-functional requirements is assembled. This list is indicative of the capabilities that the project should provide in order to accommodate these and similar use cases. Therefore, the findings presented in this deliverable are indeed guidelines that will help to efficiently design and optimize the CHARITY architecture and components.

Document revision history

Version	Date	Description of change	List of contributor(s)
v0.1	22/03/2021	First version of scenarios and requirements	HOLO3D, SRT, ORAMA, DOTES, ORBK, UTRC
v0.2	30/03/2021	Addition of components diagrams and sequence diagrams	HOLO3D, SRT, ORAMA, DOTES, ORBK, UTRC
v0.3	12/05/2021	Minor Changes	ORAMA
v0.4	22/05/2021	Addition of security/privacy subsections and Summary of Requirements section	ORAMA
v0.5	28/05/2021	Rearrangement of ToC	ORAMA
v0.54	07/06/2021	Revision	ICT-FI
v0.6	11/06/2021	Revision of ToC	ORAMA
V1.0	25/06/2021	Final editing	EURES

Acknowledgment
The research conducted by CHARITY receives funding from the European Commission H2020 program under Grant Agreement No 101016509. The European Commission has no responsibility for the content of this document.

[1] http://creativecommons.org/licenses/by-nc-nd/3.0/deed.en_US

Executive Summary

This document presents the use cases (UCs), reference scenarios and their requirements for CHARITY. The UCs, seven in total, are subdivided into three main categories, namely a) Real-time Holographic Applications, b) Immersive Virtual Training and c) Mixed Reality Interactive Applications. The CHARITY project aims to impact these research areas by providing a platform tailored to accommodate relevant applications and suitable to support their innovative features.

To be able to efficiently design the CHARITY platform, a deep understanding of these UCs functionalities is required. To accomplish this, certain aspects of every UC were presented on a weekly basis for six months, within the Scope of WP1 and T1.1. In these meetings, UC owners provided details regarding the current as well as the envisioned UC workflows that support CHARITY. A refined summary of these presentations is described in Section 2 of this deliverable. There, for each UC, a list of potential reference scenarios is also provided along with sequence diagrams that help comprehend the interactions between the UC components and the CHARITY platform. Aspects regarding the privacy and security of data, transmitted through CHARITY platform, are addressed in a separate section per UC. Moreover, the impact of deploying these UC applications within CHARITY platform is highlighted as well as the challenges imposed by this project.

To better understand these challenges and successfully tackle them, a list of potential use case functional and non-functional requirements as well as a list of general-purpose requirements was assembled. These lists, described in Section 3 of this deliverable, are indicative of the capabilities that the project should provide to accommodate the envisioned UCs and support similar ones. Therefore, these findings are guidelines that will help to effectively design and optimize the CHARITY architecture and components. This further highlights the importance of this deliverable that will be used as a reference point for the design and deployment tasks of WP2 and WP4 respectively.

The Deliverable is structured in the following fashion. Section 1 provides an overview of this deliverable. Section 2 presents an overview, the security and privacy aspects, and the reference scenarios for all UC applications. Section 3 contains the most important general-purpose requirements, as well as use case functional / non-functional requirements. Section 4, finally, concludes this deliverable.

Table of Contents

List of Figures

List of Tables

Abbreviations

API	Application Programming Interface
AR	Augmented reality
AWS	Amazon Web Services
CCU	Concurrent users
DB	Database
DoA	Description of Action
FHD	Full High Definition
GA	Grant Agreement
GPU	Graphics processing unit
HD	High Definition
HDMI	High-Definition Multimedia Interface
HLS	HTTP live streaming
HMD	Head-mounted display
JSON	Javascript Object Notation
KPI	Key Performance Indicators
LIDAR	Light Detection And Ranging
QoE	Quality of Experience
QoS	Quality of Service
RTMP	Realtime messaging protocol
RTT	Round-trip time
SaaS	Software as a service
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UC	Use Case
UDP	User Datagram Protocol
UI	User interface
URL	Universal resource locator
VM	Virtual machine
VR	Virtual reality
XR	Extended reality

1 Introduction

The CHARITY project envisions the creation of a Cloud framework for Holography and Cross Reality framework that, utilizing edge resources and devices, will host demanding applications for Virtual Reality, Augmented Reality, and Holography. CHARITY expects to deliver a working prototype that will meet the requirements of such applications and ultimately pave the way to the mass adoption of more advanced media applications. The CHARITY platform will be tested through a set of representative UCs, organized in three main categories, namely Real-time Holographic, Immersive Virtual Training, and Mixed Reality Interactive applications. To understand and overcome the challenges of the project, a thorough understanding of all UC applications is required.

This deliverable will provide UC specification details and a complete set of functional and non-functional requirements for the CHARITY framework. These will drive the CHARITY architecture design and the development of components, techniques, and algorithms.

1.1 Purpose of this document

The purpose of this document is to describe the UCs, provide reference scenarios, and requirements for the CHARITY platform. CHARITY UCs are organized into three broad categories. The first category, Real-time holographic applications, includes three UCs: Holographic concerts, Holographic meetings, and Holographic assistants. Secondly, the category of Immersive virtual training deals with two UC, namely Virtual reality medical training, and Virtual tour creator. Finally, the category of Mixed reality interactive applications consists of two UCs: Mixed reality gaming, and Aeronautical operations training simulation. For each UC, a brief description of its expected functionality is initially provided. Then, further details of the internal components and the current workflow of the application is given. To outline the interaction of the UCs with the CHARITY platform, and highlight the impact of this project, UC owners provide the envisioned workflow of their tailored application to the CHARITY platform. This document also presents with full details the problems that are solved, as well as the additional benefits and functionalities that arise by the adaptation of these UCs to the CHARITY platform. Reference scenarios provided by the UC owners help to better understand the internal workflow of a variety of high-end applications. They also provide insights regarding the capabilities that CHARITY platform must be able to provide to accommodate such applications. To this end, a list of functional, and non-functional use case requirements, as well as a list of general-purpose requirements are put together to help identify the major challenges in each UC category. Lastly, the KPIs of each UC are presented and co-related with the aforementioned requirements.

1.2 Relation to other Task and Work Packages

The content of this document is the foundation for work in all work packages to realize the CHARITY architecture solution. The deliverable defines the design and development requirements for the application to establish an adaptive and robust edge/cloud infrastructure continuum. The deliverable is strictly linked with the upcoming deliverables and Work Packages:

- T1.2 – D1.3 CHARITY architecture design and specification.
- WP2 – D2.1 Edge and cloud infrastructure resource and computational continuum orchestration system (preliminary), D2.2 Edge and cloud infrastructure resource and computational continuum orchestration system (final).
- WP3 – Drive the work on this work package.
- WP4 – Drive the work on this work package.

1.3 CHARITY approach

CHARITY aims to create a suitable cloud framework that will overcome the challenges and meet the requirements of applications in the domain of Holography, Virtual and Augmented Reality. To achieve this, a subset of such innovative media applications is analysed in this document. The adaptation of these UCs to the CHARITY infrastructure requires a deep understanding of their internal components and their interaction. As a first step, UC owners proposed the conceptual cloud/edge-based workflow of their application. After a series of weekly discussions with the experts designing the CHARITY platform, these workflows were adapted to take full advantage of the capabilities of the platform. This ongoing exchange of know-how gives better insight of what the platform should provide to the UC owners and provides the foundation of the CHARITY architecture design. Ultimately, it will seek to create the base of a better adoption of more advanced media applications by a large community of users and companies, even outside the consortium.

1.4 Structure of the document

In Section 1, an overview of this deliverable purpose is provided. Section 2 provides detailed use case descriptions, arguments on security/privacy aspects, and addresses the respective reference scenarios for all use case applications. Section 3 presents general-purpose requirements and use case specific requirements. Lastly, in Section 4, we conclude this deliverable.

2 CHARITY Use cases

This section presents the use cases that will be utilized to tackle the most demanding challenges from the advanced AR, VR and Holographic- based applications, which will ultimately allow the validation, demonstration and showcasing of the edge/cloud computing innovations of the CHARITY project. These use cases are organized in three main categories, namely, Real-time Holographic applications, Immersive virtual training, and Mixed reality interactive applications. The following table summarizes the three UC categories with the specific UCs and the involved partners.

Table 1: UC categories

UC1: Real-time Holographic applications	
UC1-1: Holographic Concerts	HOLO3D
UC1-2: Holographic Meetings	HOLO3D
UC1-3: Holographic Assistant	SRT
UC2: Immersive Virtual Training	
UC2-1: VR Medical Training	ORAMA
UC2-2: VR Tour Creator	DOTES
UC3: Mixed Reality Interactive applications	
UC3-1: Collaborative Gaming	ORBK
UC3-2: Manned-Unmanned Operations Trainer	UTRC

2.1 Use case 1 – Real-time Holographic applications

2.1.1 UC1-1 Holographic Concerts Application

2.1.1.1 Overview

AVCOM ENTERTAINMENT SRL (HOLO3D) is responsible for the Holographic Concerts use case within CHARITY. This application utilizes a 3D holographic display that creates seemingly 3D holograms from transformed 2D video streams. Figure 1 presents an overview of the use case.

The band members (referred to as *Musicians*) are situated in different geolocations, and are virtually taking part in a concert, as holograms. They are equipped with video camera, light system, black background, microphone, PC, software for live streaming, and secondary screen. Each musician's hologram is transmitted on a dedicated holographic display on stage. The band members are filmed, and their respective 2D video streams are transmitted to their assigned holographic devices. The *Audience* is situated in front of the stage, which is equipped with holographic displays, PCs, live streaming software, video camera, and microphone. The 2D stream, filmed at the stage site, is transmitted back as feedback to the band members. Figure 2 depicts the conceptual architecture of the system, simplified to two Musicians.

The Musician-PC application streams, via HTTP, the FHD/2K/4K HD recorded video stream to the CHARITY cloud service where it is transformed and streamed to the dedicated holographic display. Charity Cloud Service will be able to perform geometric transformations to the video stream (e.g., crop, resize, flip, etc.), in order to prepare it for the assigned type of holographic device (see Figure 3, Figure 4). The Client-PC receives and synchronizes the video stream with the other Musician streams and displays it in the holographic display through HDMI interface. The Audience is also filmed, and their video stream is transmitted directly to the Musicians for feedback purposes from the stage (e.g., Figure 5). This reverse process is simpler since it does not require any video transformations or synchronization.

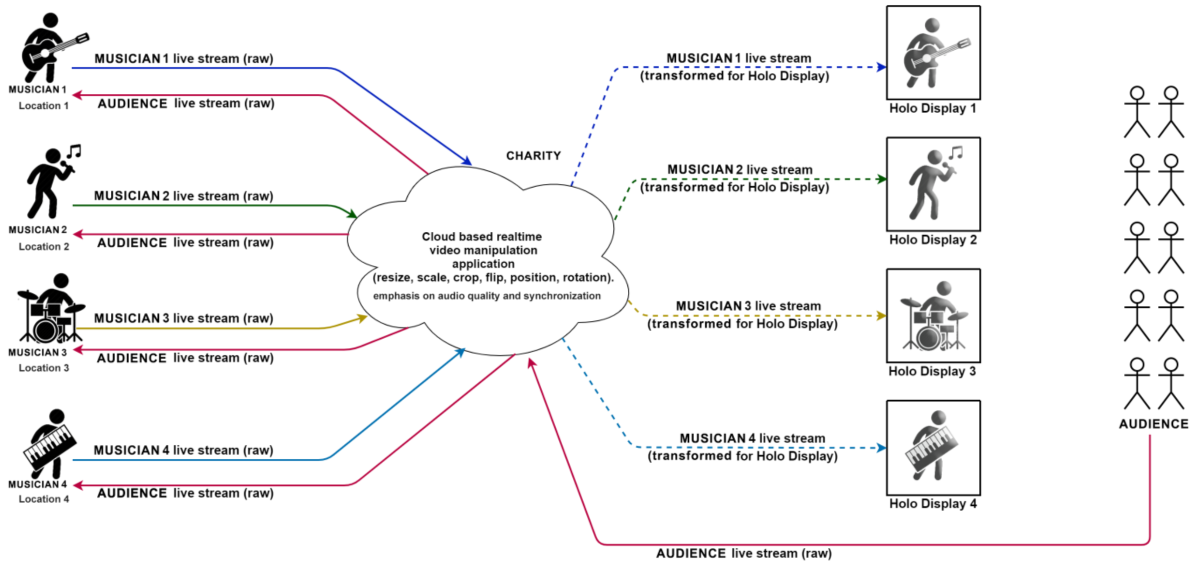

MUSICIAN 1
Location 1
MUSICIAN 1 live stream (raw)
AUDIENCE live stream (raw)
MUSICIAN 2
Location 2
MUSICIAN 2 live stream (raw)
AUDIENCE live stream (raw)
MUSICIAN 3
Location 3
MUSICIAN 3 live stream (raw)
AUDIENCE live stream (raw)
MUSICIAN 4
Location 4
MUSICIAN 4 live stream (raw)
AUDIENCE live stream (raw)
CHARITY
Cloud based realtime video manipulation application
(resize, scale, crop, flip, position, rotation).
emphasis on audio quality and synchronization
MUSICIAN 1 live stream (transformed for Holo Display)
Holo Display 1
MUSICIAN 2 live stream (transformed for Holo Display)
Holo Display 2
MUSICIAN 3 live stream (transformed for Holo Display)
Holo Display 3
MUSICIAN 4 live stream (transformed for Holo Display)
Holo Display 4
AUDIENCE
AUDIENCE live stream (raw)

Figure 1: Live Holographic Band conceptual overview.

The CHARITY service should be able to perform simple video geometric transformations: scale, position, rotation, flip and crop; these transformations are required to make the video stream compatible with the assigned holographic display.

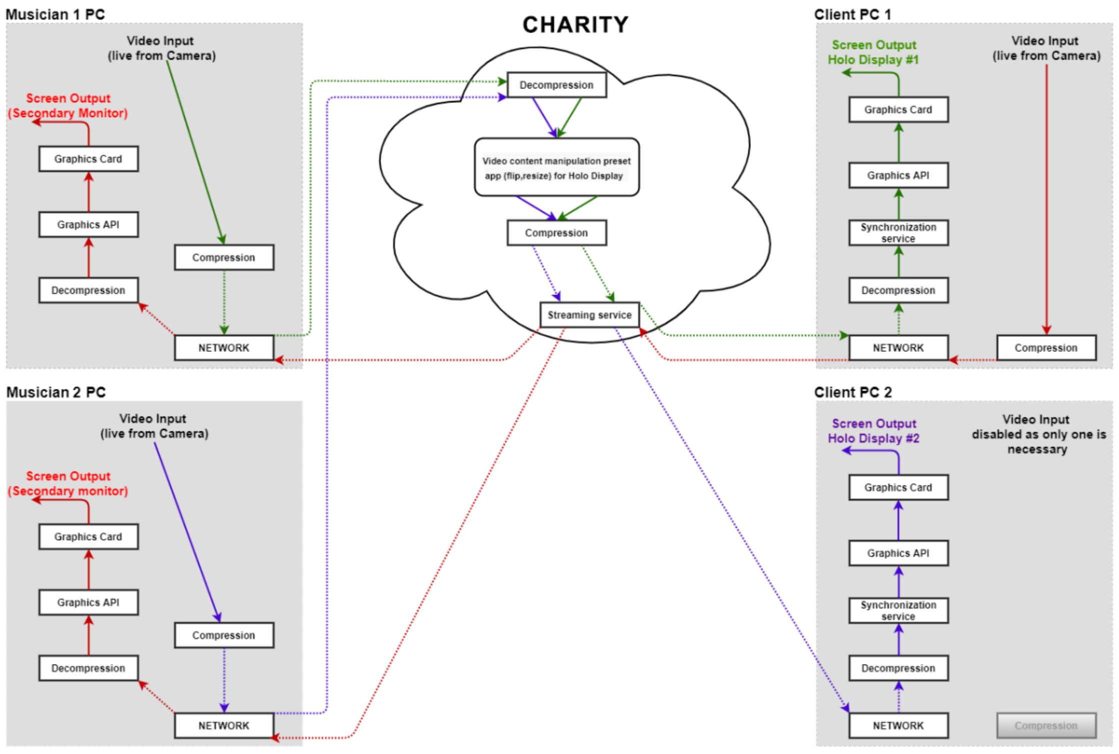


Figure 2: Live Holographic Band structural diagram.

The challenge in this use case is to synchronize all Musicians' video streams prior to displaying on the assigned holographic device. In this case, it would be worth exploring the option of separating the audio from the video stream, so that the audio (which is the most important) is still flawless in case of bandwidth drop, sacrificing video quality for audio. The overall latency between Musicians and Audience is not of primary importance, as long as the projected 2D streams from Musicians are synchronized. The synchronization will be achieved through a local client-PC's service and network time protocol (NTP). The overall latency of the already synchronized streams is a secondary priority; it can be sacrificed, as in this case sound quality and synchronization are paramount.

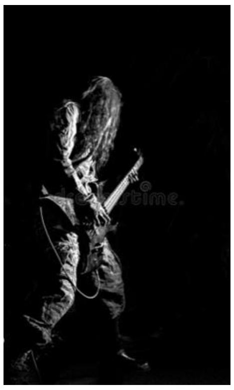


Figure 3: Example of a still frame from the Musician raw video stream.

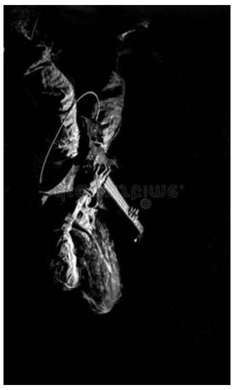


Figure 4: Example of a transformed still frame produced by CHARITY service (flip).

Figure 5: Still frame sent by the client-PC and seen by a musician.

2.1.1.2 Security/Privacy Aspects

Holographic Concerts Application use case is based on local processing supported by CHARITY services. Each client-PC is assigned to its own instance of processing modules in the CHARITY platform. A type of authentication to the CHARITY platform would be necessary. All users (Musicians and Audience) will have to consent in their filming. Video streams transmitted to/from CHARITY are encrypted using SSL/TLS. Processed data is not stored permanently within the local systems or the CHARITY service.

2.1.1.3 Reference Scenarios

2.1.1.3.1 Musician-PC Application Start Pipeline

The video input from each Musician-PC is compressed and transmitted to CHARITY cloud service where it is decompressed and transformed using a preset and rendered into a compatible format with its assigned holographic display. The rendered video is compressed and streamed to the client-PC where it is decompressed, synchronized with the other video streams and sent to the holographic display (see Figure 6).

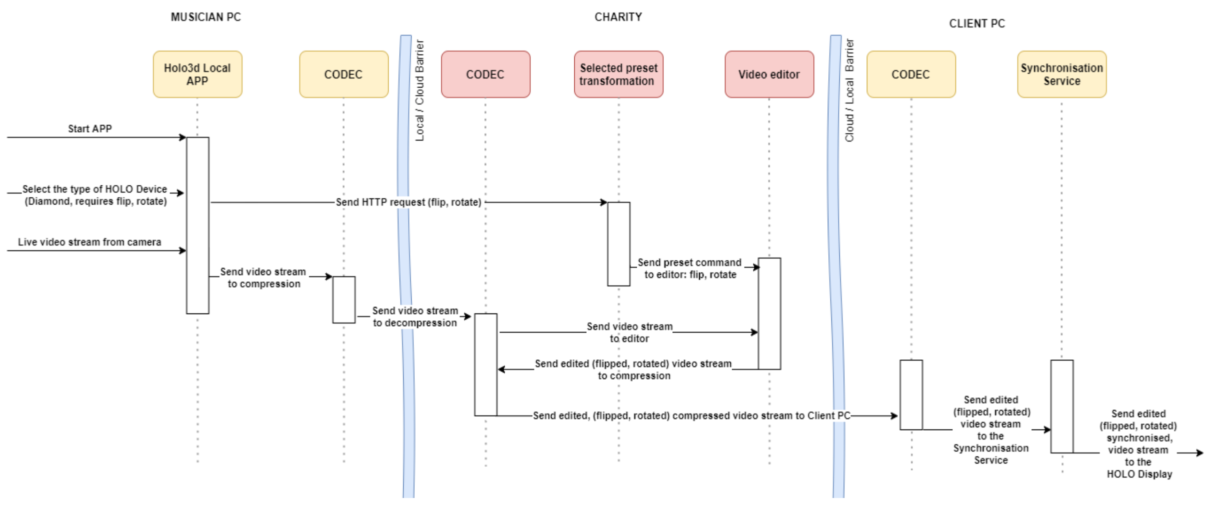

MUSICIAN PC
CHARITY
CLIENT PC
Holo3d Local APP
CODEC
Local / Cloud Barrier
CODEC
Selected preset transformation
Video editor
Cloud / Local Barrier
CODEC
Synchronisation Service
Start APP
Select the type of HOLO Device (Diamond, requires flip, rotate)
Live video stream from camera
Send HTTP request (flip, rotate)
Send preset command to editor: flip, rotate
Send video stream to compression
Send video stream to decompression
Send video stream to editor
Send edited (flipped, rotated) video stream to compression
Send edited, (flipped, rotated) compressed video stream to Client PC
Send edited (flipped, rotated) video stream to the Synchronisation Service
Send edited (flipped, rotated) synchronised, video stream to the HOLO Display

Figure 6: Sequence diagram for the scenario of application start on the Musician-PC.

2.1.1.3.2 Client-PC Application Start Pipeline

The Audience video stream does not need any synchronization or transformation and is transmitted directly to the Musician-PC for feedback (Figure 7). Video editing in terms of resolution modification may be involved.

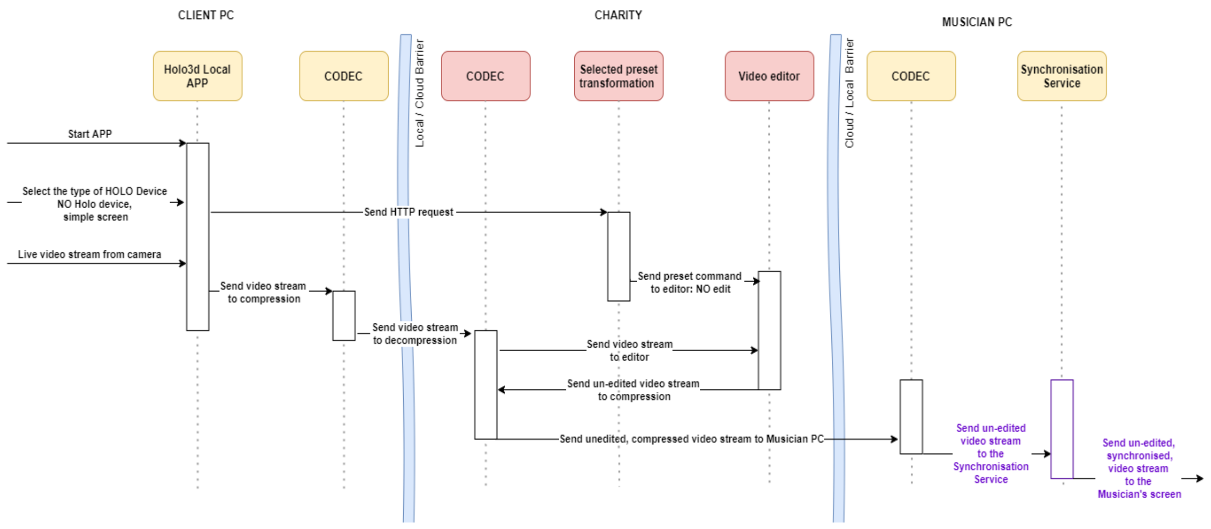

CLIENT PC
CHARITY
MUSICIAN PC
Holo3d Local APP
CODEC
Local / Cloud Barrier
CODEC
Selected preset transformation
Video editor
Cloud / Local Barrier
CODEC
Synchronisation Service
Start APP
Select the type of HOLO Device NO Holo device, simple screen
Live video stream from camera
Send HTTP request
Send preset command to editor: NO edit
Send video stream to compression
Send video stream to decompression
Send video stream to editor
Send un-edited video stream to compression
Send unedited, compressed video stream to Musician PC
Send un-edited video stream to the Synchronisation Service
Send un-edited, synchronised, video stream to the Musician's screen

Figure 7: Sequence diagram for the scenario of application start on the Client-PC.

2.1.2 UC1-2 Holographic Meetings Application

2.1.2.1 Overview

AVCOM ENTERTAINMENT SRL (HOLO3D) is also responsible for the Holographic Meetings use case within CHARITY. This application utilizes a 3D holographic display that creates seemingly 3D holograms from transformed 2D video streams.

Figure 8 depicts the current state of the Holographic meetings application where the meeting participant (*Speaker*), situated in the same geolocation as the audience, is filmed in a 2D video in front of a black background. The Speaker is equipped with video camera, light system, black background, microphone, PC, software for live streaming (Resolume Arena), and a secondary screen. The Audience is equipped with a holographic device, PC, software for live streaming, video camera, microphone. The Speaker-PC transforms the 2D video in real-time and renders it accordingly to be projected on the holographic display (e.g., Figure 12, Figure 13, and Figure 14). The resulting video is transmitted to the holographic display via an HDMI cable.

The current setup is able to transmit video to many, single-type, holographic devices situated in the same geolocation. This setup is limited though by the length of the HDMI cable and therefore it is not able to transmit the rendered video to different geolocations or to various types of holographic displays.

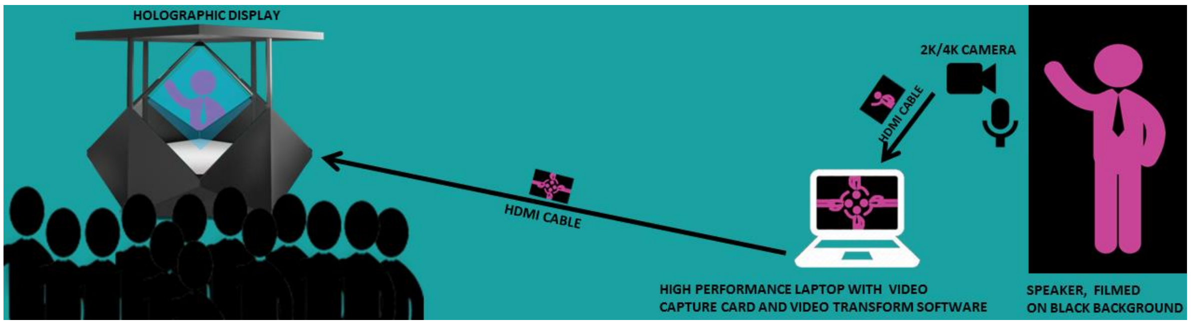

HOLOGRAPHIC DISPLAY
2K/4K CAMERA
HDMI CABLE
HDMI CABLE
HIGH PERFORMANCE LAPTOP WITH VIDEO CAPTURE CARD AND VIDEO TRANSFORM SOFTWARE
SPEAKER, FILMED ON BLACK BACKGROUND

Figure 8: The current conceptual overview of Holographic Meetings.

The objective of this cloud application is to enable the Speaker to be situated at any geolocation and transmit its video to numerous holographic displays in various venues concurrently. Figure 9 presents an overview of the use case. In this setup, the Speaker-PC captures and transmits the 2D video to a CHARITY service that transforms and renders it accordingly for any type holographic display connected to CHARITY (e.g., Figure 12, Figure 13, and Figure 14). The Speaker-PC may receive, un-edited video feed (e.g., Figure 15), filmed at the audience geolocation, to enable visual communication between the Speaker and the Audience in real-time. Communication is therefore possible between various Speakers in the same venue. Furthermore, the Speaker will be able to share visual content with the audience.

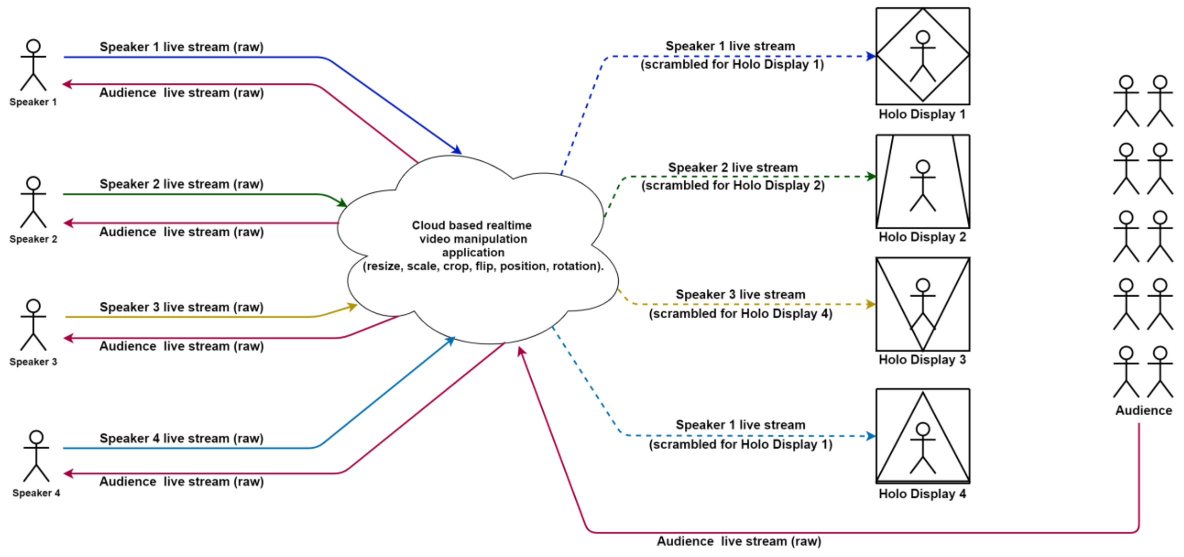

Speaker 1
Speaker 1 live stream (raw)
Audience live stream (raw)
Speaker 2
Speaker 2 live stream (raw)
Audience live stream (raw)
Speaker 3
Speaker 3 live stream (raw)
Audience live stream (raw)
Speaker 4
Speaker 4 live stream (raw)
Audience live stream (raw)
Cloud based realtime video manipulation application (resize, scale, crop, flip, position, rotation).
Speaker 1 live stream (scrambled for Holo Display 1)
Holo Display 1
Speaker 2 live stream (scrambled for Holo Display 2)
Holo Display 2
Speaker 3 live stream (scrambled for Holo Display 4)
Holo Display 3
Speaker 1 live stream (scrambled for Holo Display 1)
Holo Display 4
Audience
Audience live stream (raw)

Figure 9: Holographic Meeting conceptual overview.

Figure 10, depicts the conceptual architecture of the application. Each Speaker-PC application streams, via HTTP, the FHD/2K/4K HD recorded video stream to the CHARITY cloud service where it is transformed and streamed to the dedicated holographic display. CHARITY service will be able to perform geometric transformations to the video stream (position, rotate, crop, resize, flip, etc.), in order to prepare it for the assigned type of holographic device. The Speaker may share visual content (jpg, mp4, ppt, pdf) which is converted by the Speaker-PC application to video stream, replacing the camera video stream. The Client-PC receives and displays the video stream in the holographic display through HDMI interface. The Audience is also filmed, and their video stream is transmitted directly to the Speaker for feedback purposes from the venue (e.g., Figure 15). This reverse process is simpler since it does not require any video transformations.

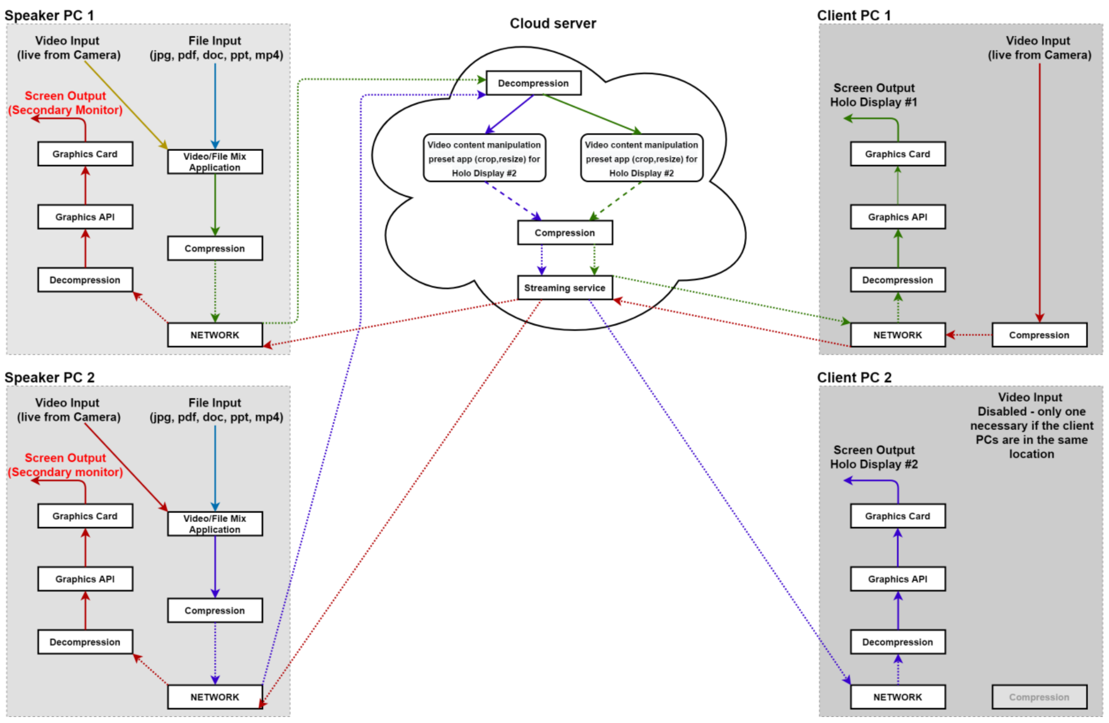

Speaker PC 1
Video Input (live from Camera)
File Input (jpg, pdf, doc, ppt, mp4)
Screen Output (Secondary Monitor)
Graphics Card
Video/File Mix Application
Graphics API
Compression
Decompression
NETWORK
Cloud server
Decompression
Video content manipulation preset app (crop,resize) for Holo Display #2
Video content manipulation preset app (crop,resize) for Holo Display #2
Compression
Streaming service
Client PC 1
Video Input (live from Camera)
Screen Output Holo Display #1
Graphics Card
Graphics API
Decompression
NETWORK
Compression
Speaker PC 2
Video Input (live from Camera)
File Input (jpg, pdf, doc, ppt, mp4)
Screen Output (Secondary monitor)
Graphics Card
Video/File Mix Application
Graphics API
Compression
Decompression
NETWORK
Client PC 2
Video Input Disabled - only one necessary if the client PCs are in the same location
Screen Output Holo Display #2
Graphics Card
Graphics API
Decompression
NETWORK
Compression

Figure 10: Holographic Meeting structural diagram.

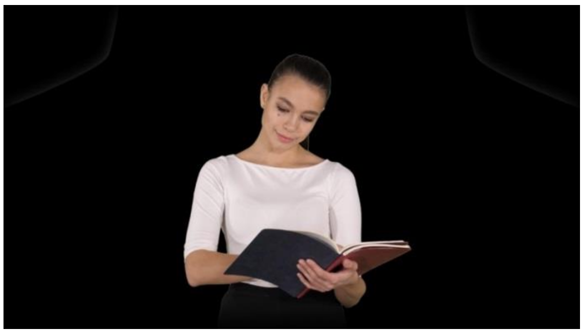


Figure 11: Still frame of un-edited video stream.

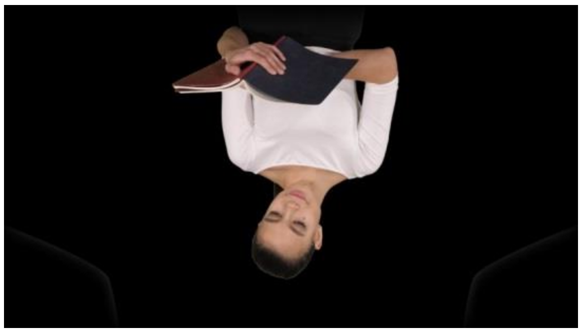


Figure 13: Still frame of edited video stream (Flip).

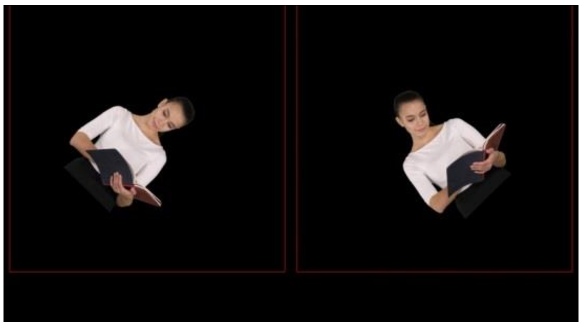


Figure 12: Still frame of edited video stream (position, size, flip).

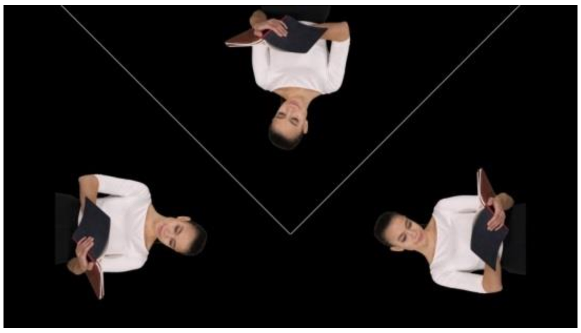


Figure 14: Still frame edited video stream (position, size, flip, rotation).

The CHARITY service should be able to perform simple video geometric transformations: scale, position, rotation, flip and crop; these transformations are required to make the video stream compatible with the assigned holographic display.

Even though Holographic Meetings (UC1-2) and Holographic Concerts (UC1-1) will be using the same local application and CHARITY services, they differ in two main aspects: (a) the synchronization service is not necessary for UC1-2, (b) the Speakers can share visual content with the Audience in UC1-2.

Figure 15: Still frame of the Audience.

2.1.2.2 Security/Privacy Aspects

Holographic Concerts Application use case is based on local processing supported by CHARITY services. Each client-PC is assigned to its own instance of processing modules in the CHARITY platform. A type of authentication to the CHARITY platform would be necessary. All users (Speakers and Audience) will have to consent in their filming. Video streams transmitted to/from CHARITY are encrypted using SSL/TLS. Processed data is not stored permanently within the local systems or the CHARITY service.

2.1.2.3 Reference Scenarios

2.1.2.3.1 Speaker-PC Application Start Pipeline

The video input from the Speaker-PC is compressed and transmitted to CHARITY cloud service where it is decompressed and transformed using a preset and rendered into a compatible format with its assigned holographic display. The rendered video is compressed and streamed to the client-PC where it is decompressed and sent to the holographic display (see Figure 16).

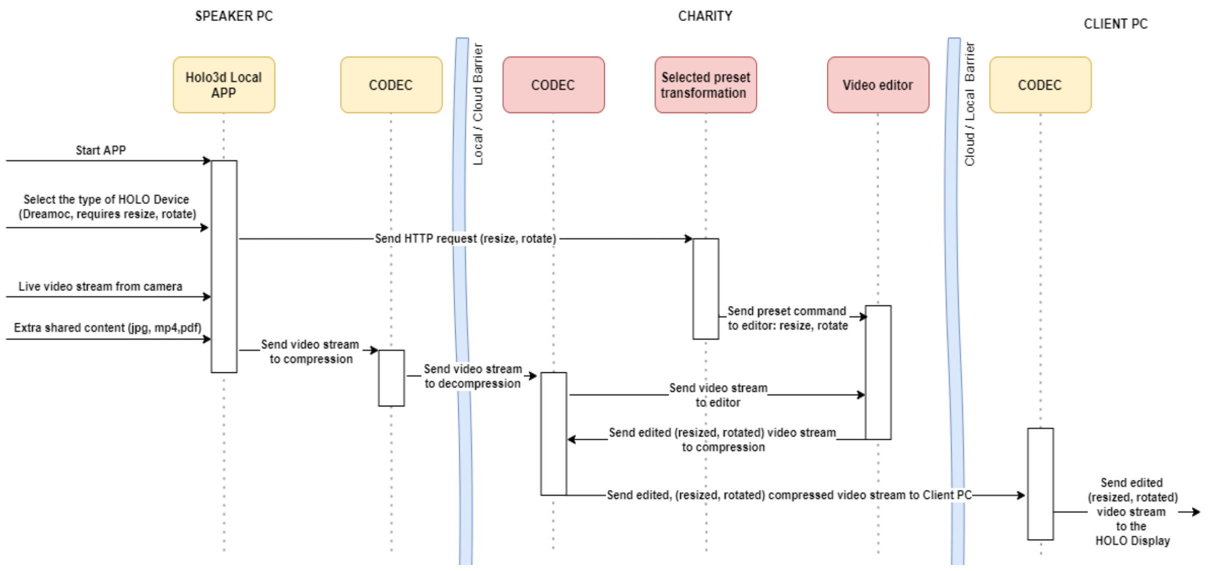

SPEAKER PC
CHARITY
CLIENT PC
Holo3d Local APP
CODEC
Local / Cloud Barrier
CODEC
Selected preset transformation
Video editor
Cloud / Local Barrier
CODEC
Start APP
Select the type of HOLO Device (Dreamoc, requires resize, rotate)
Send HTTP request (resize, rotate)
Live video stream from camera
Send preset command to editor: resize, rotate
Extra shared content (jpg, mp4,pdf)
Send video stream to compression
Send video stream to decompression
Send video stream to editor
Send edited (resized, rotated) video stream to compression
Send edited, (resized, rotated) compressed video stream to Client PC
Send edited (resized, rotated) video stream to the HOLO Display

Figure 16: Sequence diagram of scenario application start on the Speaker-PC.

2.1.2.3.2 Client-PC Application Start Pipeline

The Audience video stream does not need any transformation and is transmitted directly to the Speaker-PC for feedback (Figure 17).

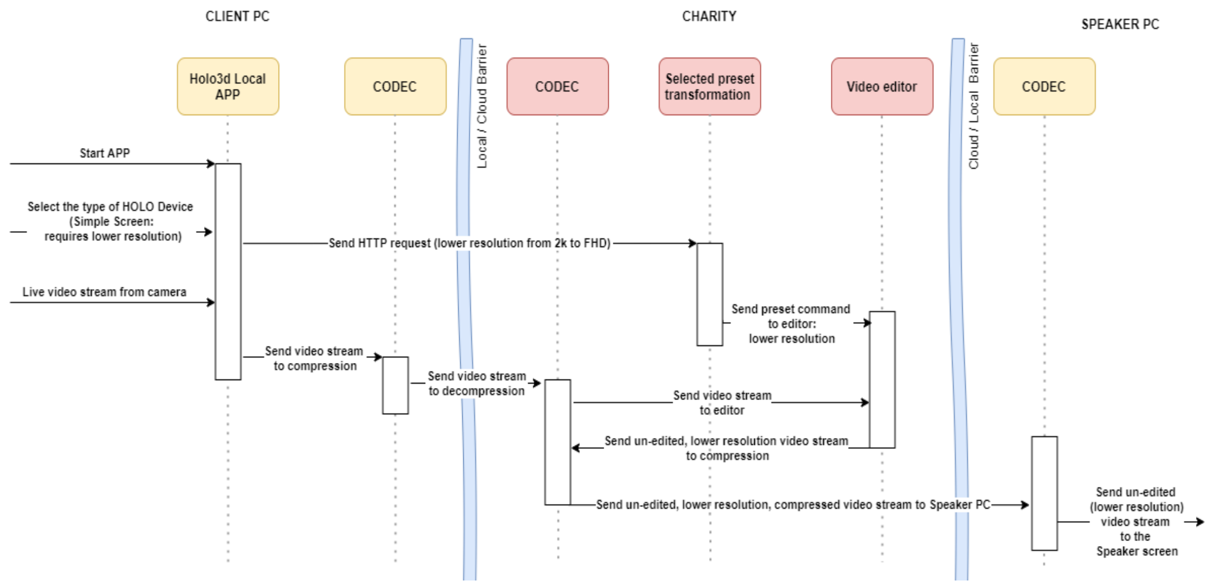

CLIENT PC
CHARITY
SPEAKER PC
Holo3d Local APP
CODEC
Local / Cloud Barrier
CODEC
Selected preset transformation
Video editor
Cloud / Local Barrier
CODEC
Start APP
Select the type of HOLO Device (Simple Screen: requires lower resolution)
Send HTTP request (lower resolution from 2k to FHD)
Live video stream from camera
Send preset command to editor: lower resolution
Send video stream to compression
Send video stream to decompression
Send video stream to editor
Send un-edited, lower resolution video stream to compression
Send un-edited, lower resolution, compressed video stream to Speaker PC
Send un-edited (lower resolution) video stream to the Speaker screen

Figure 17: Sequence diagram of the application start on the Client-PC scenario.

2.1.3 UC1-3 Holographic Assistant Application

2.1.3.1 Overview

SeeReal Technologies GmbH (SRT) from Dresden/Germany is responsible for the Holographic Assistant use case within CHARITY. We propose a visual system where a three-dimensional (3D) avatar appears on a holographic 3D display to offer assistance for day-by-day tasks, such as answering questions or presenting various information compiled from information sources on the internet. All assistant competences will be directly supported by cloud-based services from CHARITY project or by cloud services of other companies or organizations. The holographic assistant, visualized as a 3D-avatar on a holographic 3D display, will recognize and interpret speech and answers in natural English language (see Figure 18). In addition, contextualized information will be visualized.

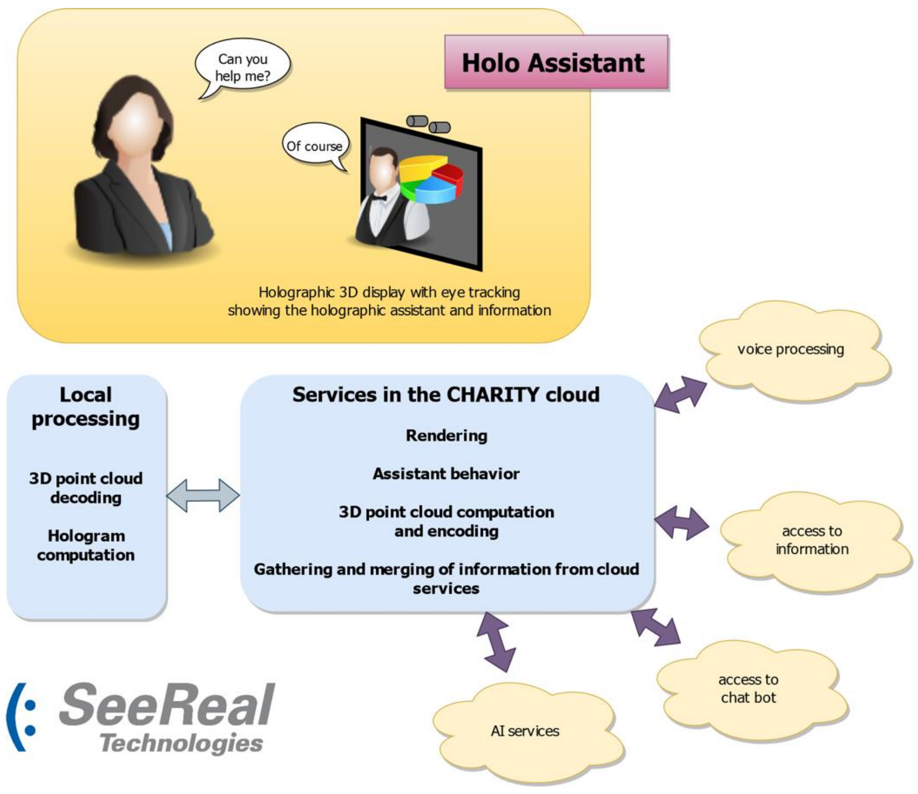

Can you help me?
Of course
Holo Assistant
Holographic 3D display with eye tracking showing the holographic assistant and information
Local processing
3D point cloud decoding
Hologram computation
Services in the CHARITY cloud
Rendering
Assistant behavior
3D point cloud computation and encoding
Gathering and merging of information from cloud services
voice processing
access to information
access to chat bot
AI services
SeeReal Technologies

Figure 18: Conceptual overview of the holographic assistant.

The Holographic Assistant use case is based on software and hardware supported by CHARITY cloud services. Some modules and services will run in the CHARITY cloud - i.e., rendering, assistant logic and 3D point cloud processing. The assistant software thus runs independently from capabilities and resources of the end user device. Information and services which the holographic assistant will offer are accessed through cloud-based services via APIs and interfaces provided by these information providers. The advantage of this approach is that the end user device only needs to visualize the resulting 3D data received as a streamed 3D point cloud. All gathering, processing of information and rendering, compression of 3D data are offloaded to the cloud. Bandwidth and computation requirements are lower since only the processed results and assistant visual/audio information needs to be transferred (especially important for mobile devices).

The display technology used for visualization will be based on real 3D Holography, a true interference-based technology (cf. interference of light) where the user can see depth in space (true 3D) and can focus the details of the 3D-avatar or 3D-information shown in space with his eyes (like with camera changing focus). Overall, this type of display provides the same natural viewing experience for the user's eyes as if the holographic assistant would be a real object or person. The additional information visualized could be text data (written information), image data (photos, maps etc.) or 3D-data (3D-models, 3D-visualizations).

SRT is starting from scratch for developing this use case, so there is no existing application to be made cloud native. But the impact with CHARITY would be the ability to render 3D data in the cloud and stream this data in the form of a 3D point cloud (containing special information for enabling true holographic 3D) from the cloud to a Holographic 3D device. SRT is quite confident that this was not ever done before in such a constellation. In the planned scenario within CHARITY, services provided by the holographic assistant would be reduced to some simple cases/services, like weather information, information about stocks and a connection to an already existing chat bot service. Figure 19Figure shows the interactions of the software modules to enable this use case.

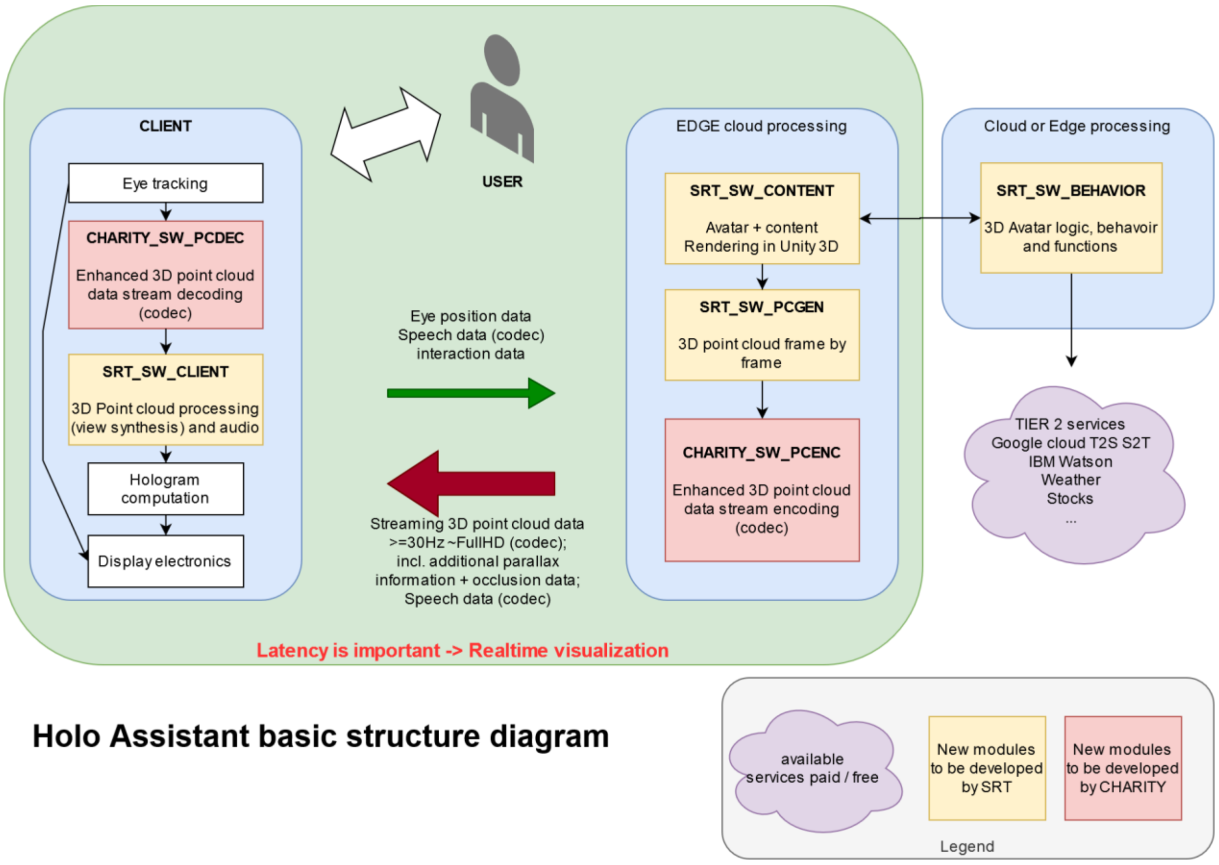

CLIENT
Eye tracking
CHARITY_SW_PCDEC
Enhanced 3D point cloud data stream decoding (codec)
SRT_SW_CLIENT
3D Point cloud processing (view synthesis) and audio
Hologram computation
Display electronics
USER
Eye position data
Speech data (codec)
interaction data
Streaming 3D point cloud data >=30Hz ~FullHD (codec); incl. additional parallax information + occlusion data; Speech data (codec)
EDGE cloud processing
SRT_SW_CONTENT
Avatar + content Rendering in Unity 3D
SRT_SW_PCGEN
3D point cloud frame by frame
CHARITY_SW_PCENC
Enhanced 3D point cloud data stream encoding (codec)
Cloud or Edge processing
SRT_SW_BEHAVIOR
3D Avatar logic, behavoir and functions
TIER 2 services
Google cloud T2S S2T
IBM Watson
Weather
Stocks
...
Latency is important -> Realtime visualization
Holo Assistant basic structure diagram
available services paid / free
New modules to be developed by SRT
New modules to be developed by CHARITY
Legend

Figure 19: Holographic Assistant structural diagram.

2.1.3.2 Security/Privacy Aspects

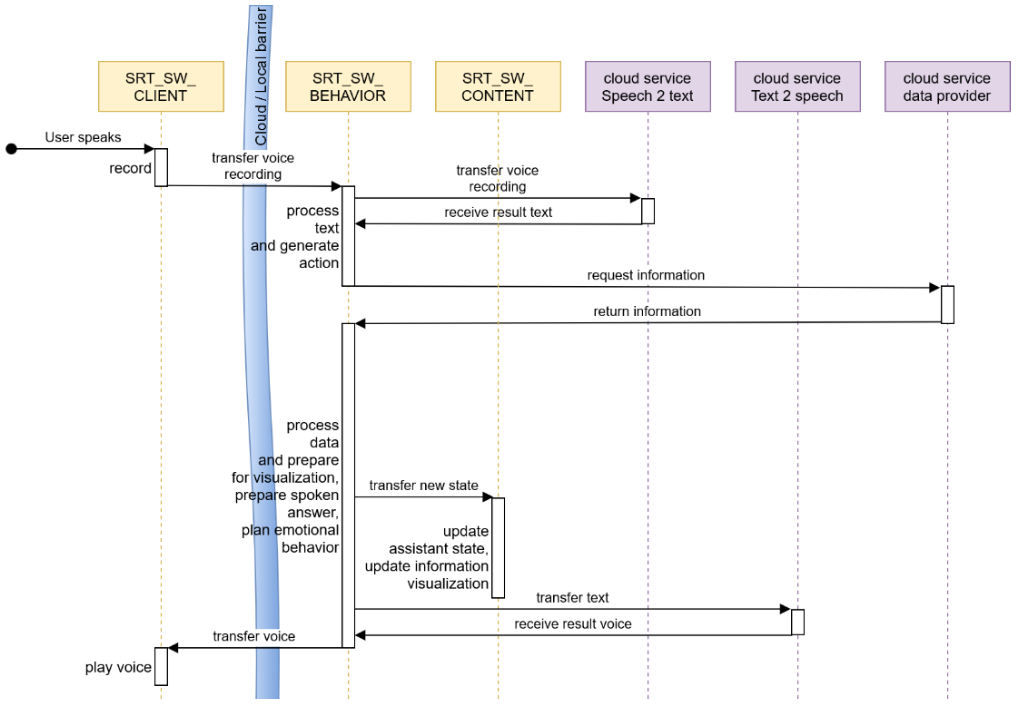
The Holographic Assistant use case is based on software and hardware supported by CHARITY- and cloud services. Each client, running local software modules, get its own instances of processing modules on the CHARITY platform assigned. Accordingly, a principal authentication against the CHARITY cloud is necessary to prevent unauthorized access. All data transferred from/to the cloud software modules are completely anonymized and do not contain any personal information about the USER - there is even no personal information requested or required. To maintain privacy aspects regarding transfer and processing of speech data, the user needs to verbally consent to this in context of a live demonstration or may sign a consent form in case of long-term use by itself. Voice data from the USER to be processed in the cloud is secured by means of encryption using state of the art methods like SSL/TLS. All other data like eye-tracking coordinates, 3D point cloud data, QoE control etc. do not need to be secured, but will be encrypted on demand. All data processed within the software modules are generally not permanently stored. This especially applies to data like voice recording or camera images processed by eye-tracking software. Eye-tracking is working only locally on the client system; only the resulting 3D eye-coordinates are transferred to the edge.

2.1.3.3 Reference scenarios

2.1.3.3.1 Asynchronous processing a question and generating an answer

In this scenario, USER asks a question, request is processed, resulting behavior and speech of assistant is computed and visualized/presented; optionally additional information is shown, depending on the type of information requested.

Figure 20 gives an overview of what happens when the USER makes a request. Basically, the request is recorded, sent to the cloud, and processed there to generate an action. This action is executed by gathering the appropriate information on the internet. The answer is finally played back. It shall be noted that the information to be visualized in context of this request are generated in a different process.



.

Figure 20: Sequence diagram for asynchronous assistant reaction to spoken input scenario.

2.1.3.3.2 Synchronous content rendering and transmission

This scenario explains the continuous and synchronous generation of content, generating 3D point cloud data, encoding in CHARITY cloud, decoding and visualization on client. Figure 21 depicts this scenario in detail. Based on eye-position data from the client, current assistant behavior data (like emotions) and information to visualize, rendering of content is updated. 3D point cloud data is generated, encoded/compressed, and transferred to client. Finally, it is decoded/decompressed and visualized.

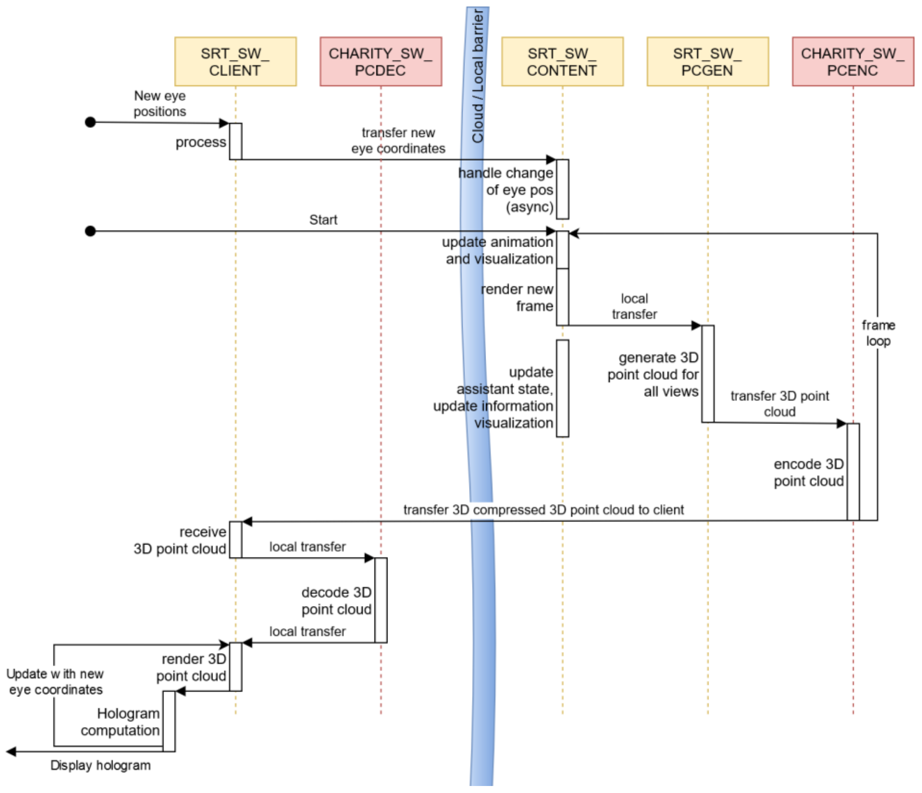


Figure 21: Sequence diagram for synchronous content rendering and transmission scenario.

2.2 Use case 2 – Immersive Virtual Training

2.2.1 UC2-1 VR Medical Training Application

2.2.1.1 Overview

Immersive collaborative VR training tools and applications must look and feel realistic. In such VR training environments, moving objects as well as transformations and deformations of soft-body 3D objects require a significant amount of processing power for mathematical calculations. The ORamaVR (ORAMA) software platform, called the MAGES SDK, is a multi-player engine that works on top of standard networking protocols. Via a novel interpolation engine running under-the-hood, we may perform virtual character animation by sharing fewer key-frames across the network and locally calculating all intermediates thus minimizing data transfer and compressing broadcasted data. The integrated cloud-based analytics engine, with user assessment and progress monitoring, captures several hundreds of events per second. Its gamified geometric algebra supports fast interpolation, animation, deformation of soft-body objects in 3D.

Collaboration between a large number of remote concurrent users (CCUs) may be accomplished by adopting a networking functionality beyond the standard client-server model where a relay server transfers the current state of deformable objects. Current solution designs are to be scaled to edge/cloud resources, facilitating advanced processing capabilities with low latency that reduce the imposed constraints on limited resources, GPU, battery, and mobility on untethered head-mounted displays (HMDs). To this end, the dependence on local heavy technology assets may be decreased.

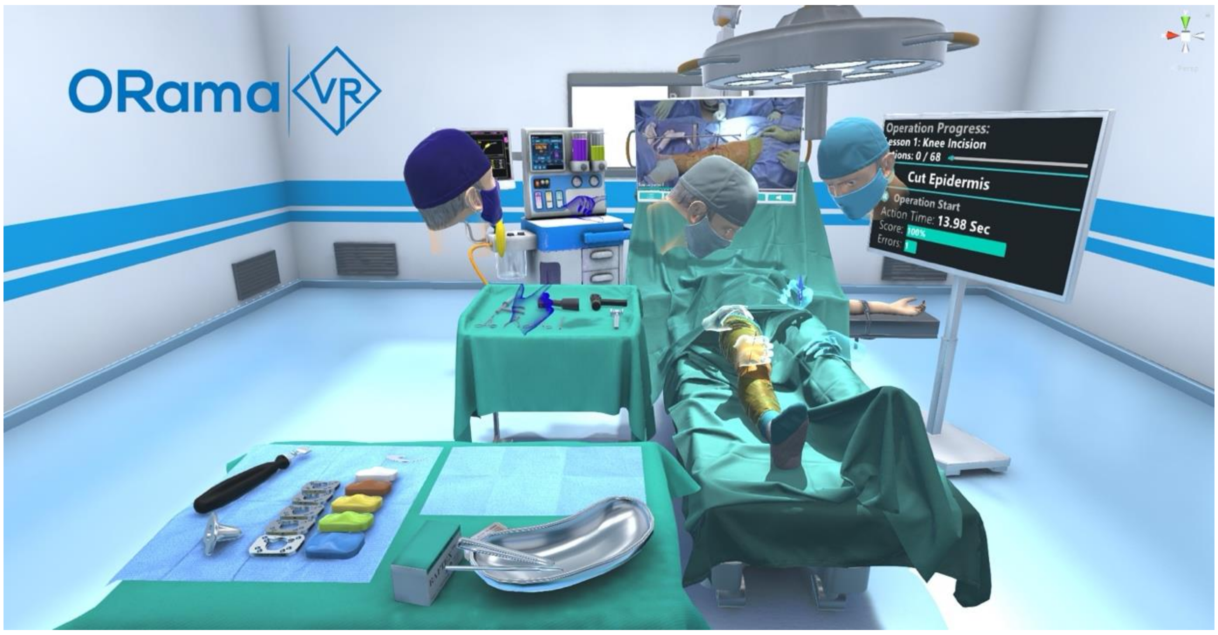


Figure 22: ORAMA's VR medical training application.

Currently, ORAMA's application (see Figure 22) is not running on the cloud. The ORAMA software platform will exploit different data services within CHARITY framework provided by available resources across the cloud-edge compute continuum. Three different aspects will be considered: a) computation offloading to nearby edge resources, leaving the mobile device responsible for UI, input/output, thus streaming only necessary content to client HMDs; b) provide a relay server functionality based on open source networking API, hosted in the cloud to handle game logic and broadcast messages to remote clients; thus it ensures low latency exchange by reducing the dependency on local services; c) run-time adaptation and dynamic optimization of the Geometric Algebra Interpolation Engine based on the network characteristics.

The adapted UC workflow adopts the micro-service concept as the main application will be deployed as virtual machine (VM) stateful micro-service on edge (see Figure 23). One instance of the application will be deployed for each HMD. The application instance will be responsible of computing, rendering, and encoding the images that will be transmitted to the HMD by a signalling server. The lightweight HMDs will be responsible for decoding and projecting the transferred images from the edge, and to capture and transfer user events (positioning, rotations, etc.) to the application instance. Current research activities focus on how to separate the physics component from the main application as a separate VM micro-service.

2.2.1.2 Security/Privacy Aspects

ORAMA product users are associated with user accounts, distinguishable on a role-based model, i.e., admins, supervisors, and members. The accounts are part of the internal ORAMA cloud services. User registration data and credentials are stored in ORAMA's portal during registration. Sensitive information of user accounts is hashed using the SHA256 algorithm and traffic is encrypted using the AES-256 specification. To this end, no data is stored locally on the HMD or on edge. The ORAMA Analytics engine uses a cloud-based user assessment service to track, monitor and present important feedback (user scores and action logs during training sessions) regarding each gamified operation. These data are anonymized and stored in the Microsoft Azure Blob storage system, utilizing its security, and are accessible in the form of JSON files through the analytics API. The later conforms with the REST service design.

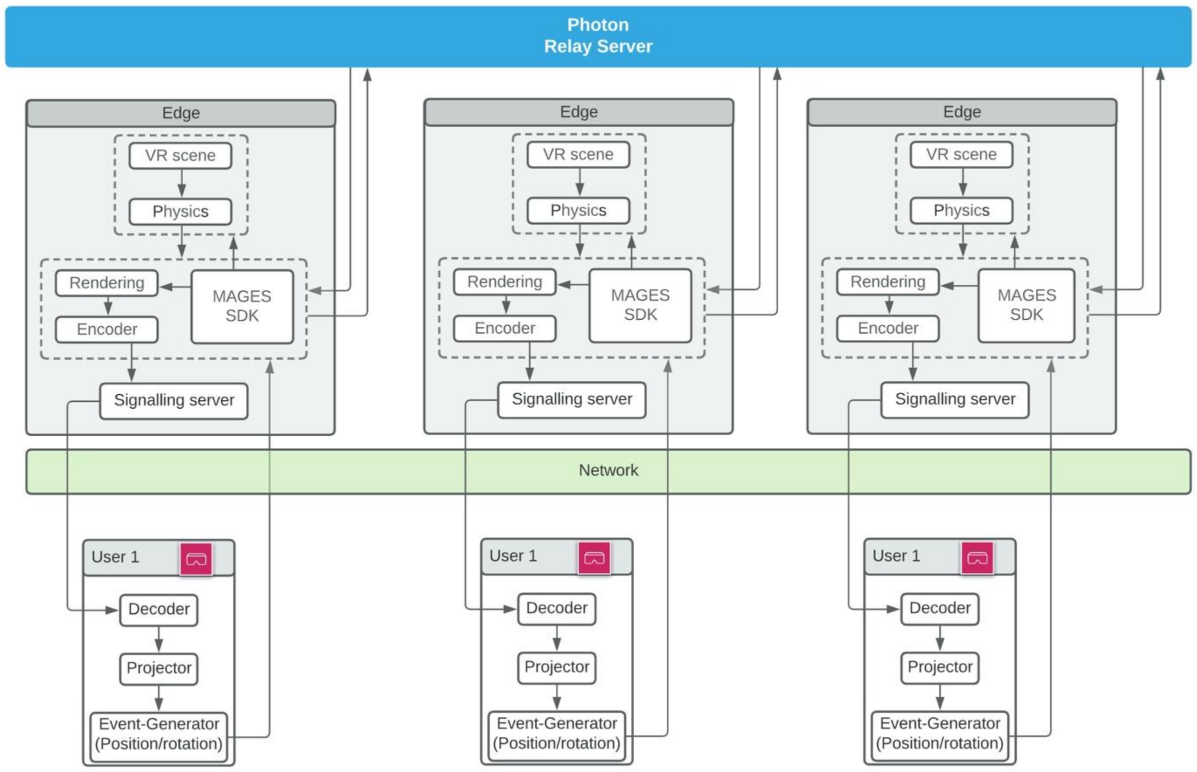


Figure 23: ORAMA's UC structural diagram.

2.2.1.3 Reference scenarios

Prerequisites for all scenarios:

- Communication to 5G or Wi-Fi is established via the HMD.
- The channel for communication with CHARITY platform should be known to the application running on the HMD.
- The channel for communication between application components will be managed and deployed by the CHARITY platform. For direct app-to-app communication, the UDP protocol is used.

2.2.1.3.1 Submission of the application

The application developer enters the CHARITY Platform portal and submits the required data to the CHARITY platform to enable the deployment of the application.

2.2.1.3.2 Start application

The user starts the application from the HMD, which initiates a launch command to the CHARITY platform, through the event bus. The CHARITY platform deploys the two application images on the closest possible edge (LSpart_1 and LSpart_2) (see Figure 24). The application exploits the open-source cross-platform WebRTC for Unity package for streaming rendered images from the LSpart_1 component to the HMD. The WebRTC is involved with a third-party signalling server residing on edge.

2.2.1.3.3 Create / Join Session

The user accesses the interface for session list on the HMD and requests through the CHARITY platform the Relay server address. The CHARITY takes care of the necessary deployment steps, if necessary, of the Photon relay server instance. Information is sent back to the LS and when connection is directly established with the relay server instance, the user retrieves the list of active sessions and requests to create a new or join an existing session in which other users may connect to (see Figure 25, Figure 26).

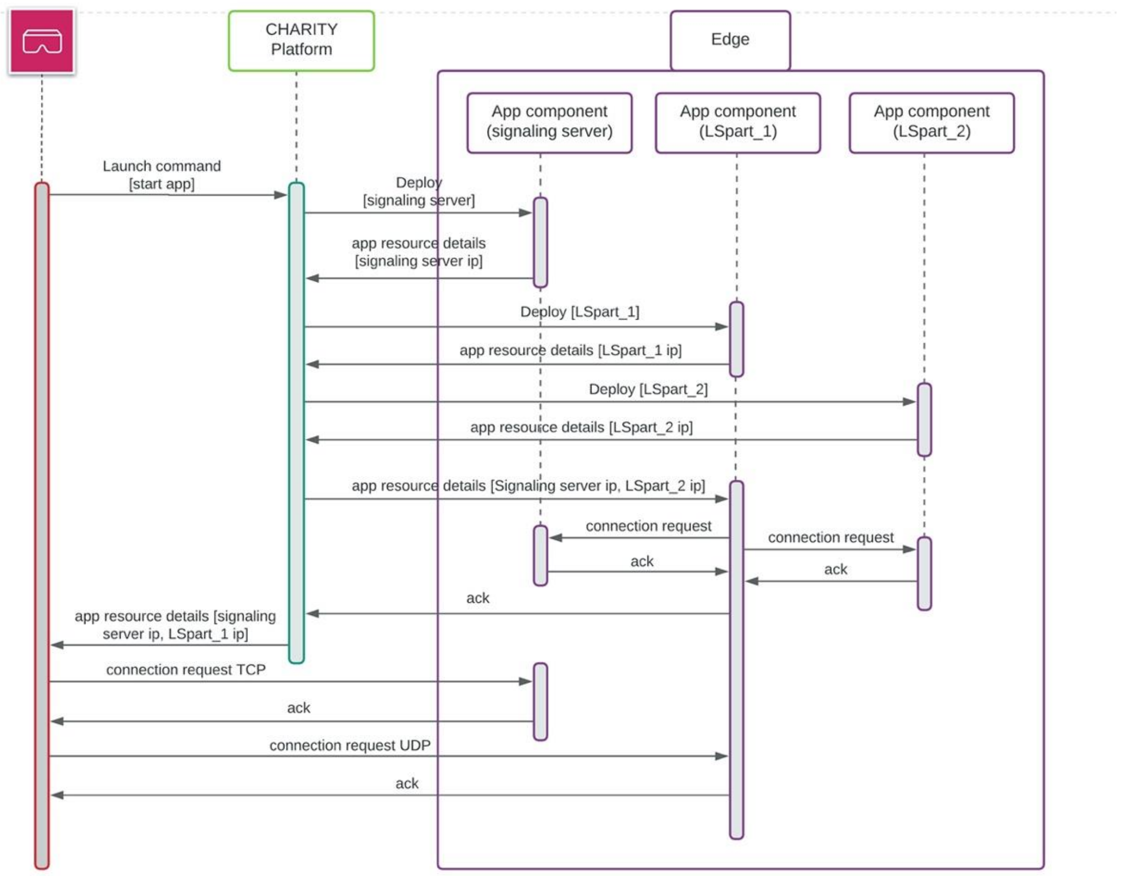

CHARITY Platform
Edge
App component (signaling server)
App component (LSpart_1)
App component (LSpart_2)
Launch command [start app]
Deploy [signaling server]
app resource details [signaling server ip]
Deploy [LSpart_1]
app resource details [LSpart_1 ip]
Deploy [LSpart_2]
app resource details [LSpart_2 ip]
app resource details [Signaling server ip, LSpart_2 ip]
connection request
connection request
ack
ack
ack
app resource details [signaling server ip, LSpart_1 ip]
connection request TCP
ack
connection request UDP
ack

Figure 24: Sequence diagram for Start application scenario.

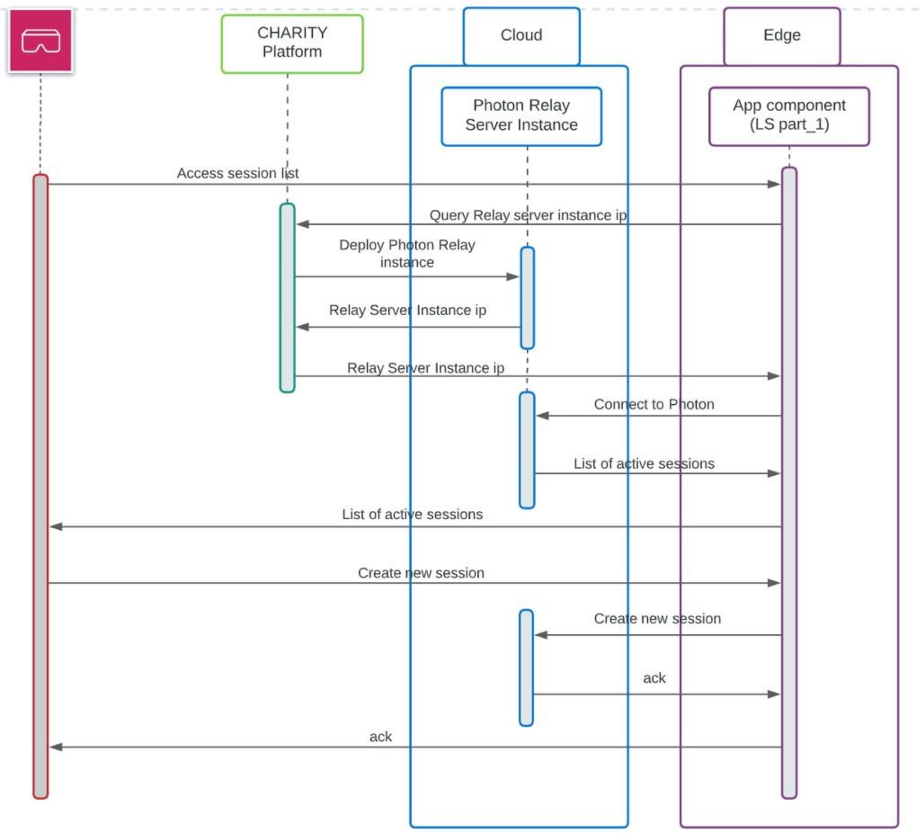


Figure 25: Sequence diagram for Create Session scenario.

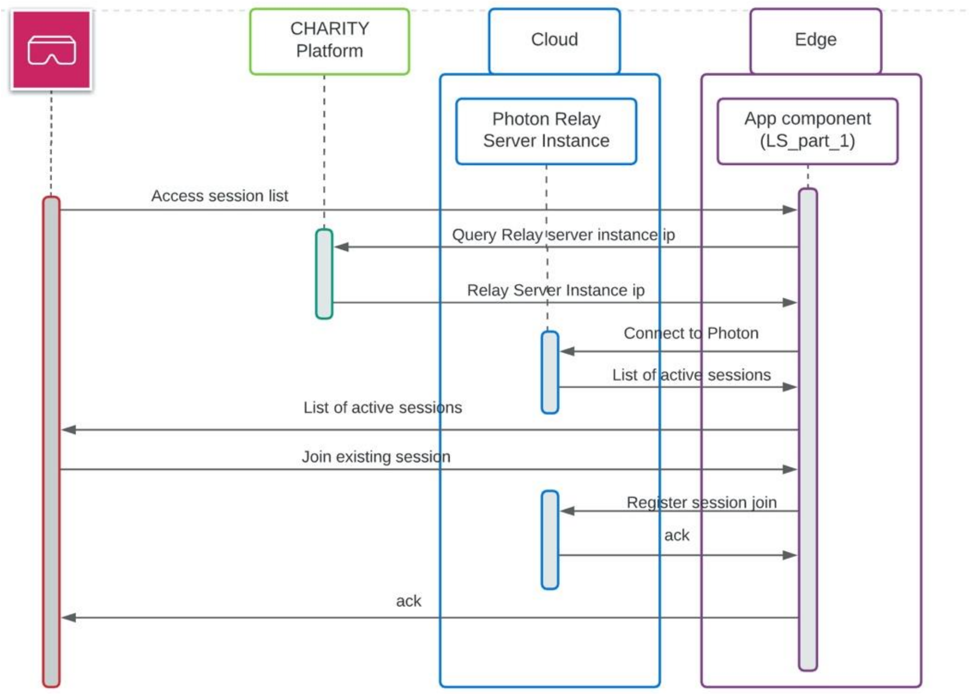


Figure 26: Sequence diagram for Joining Session scenario.

2.2.2 UC2-2 VR Tour Creator Application

2.2.2.1 Overview

Cyango is the name of Dotesfera's (DOTES) product on CHARITY. Cyango is a virtual tours software that allows anyone to build interactive experiences in Virtual Reality. It can be used either to create a virtual tour or to create interactive live streaming scenes where people can talk in real-time with all the spectators. The virtual tour can support 360 videos, panoramas, 3D models, standard images and videos and basic 3D meshes. Our use case business model works similarly to the SaaS (software as a service) model, that is, the user starts a 30-day free trial and can then upgrade to a monthly subscription to continue using the software as is.

The platform consists of two front-ends:

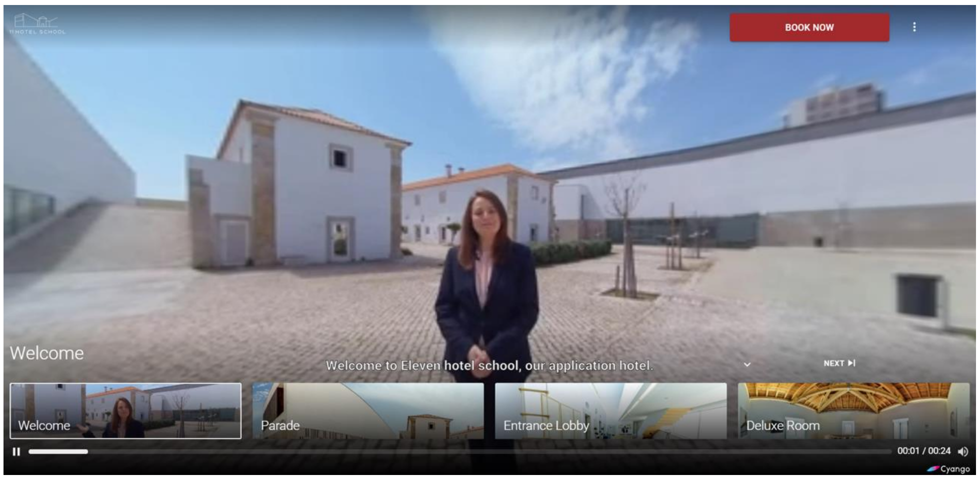


Figure 27: Story Front-end.

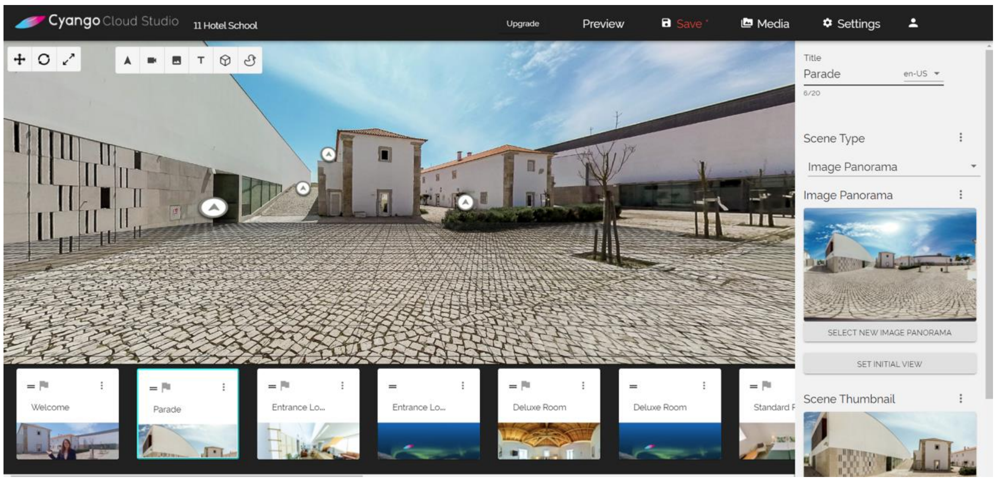


Figure 28: Cloud Studio Front-end.

- The Story front-end is where the VIEWER stakeholder can consume and experience the tour (see Figure 27).
- The Cloud Studio (Figure 28) front-end is where the USER stakeholder can create the tour, by uploading media files and using the tool on the cloud. The USER may need to use the Cloud Studio through a mobile or a low bandwidth network and must be able to upload any type of files (videos and images mostly) to the cloud. Both of these front-ends use React.js [2] and A-frame wrapper[3] with three.js[4] framework based on WebGL. A-frame uses WebXR API[5].

The back-end is where we have our API that connects to a non-SQL database (MongoDB), and it is

[2] https://reactjs.org/

[3] https://aframe.io/

[4] https://threejs.org

[5] https://developer.mozilla.org/en-US/docs/Web/API/WebXR _Device_API

written with Node.js. The current state of the platform is described as follows:

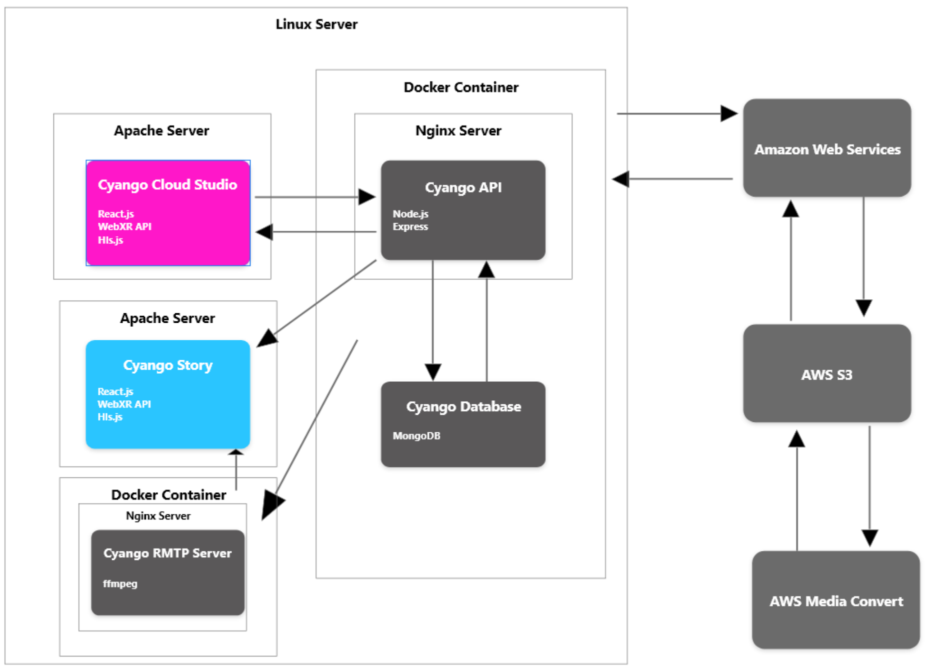


Figure 29 : Current Structural diagram of DOTES application.

The current platform has the two front-ends (Story and Cloud Studio) communicating with the backend (API) and database. Each front-end and backend service is in its Docker container with binding volumes to assure permanent data. Each time a video file gets uploaded on Cloud Studio, it triggers an AWS lambda job that converts the file to HLS format so it can be streamed. This is referred in Figure 29 as the AWS S3 and AWS Media Convert. The RTMP Server is the service component that we use when a USER wants to start live-streaming by just assigning a new RTMP URL.

The future state of the platform we envision with the contribution of CHARITY is described in Figure 30. We plan to adapt the micro-service structure, containerizing each service separately. With CHARITY, we expect to have a file host and deliver media files faster on the cloud and to render 3D models and post-process videos. That is the goal of one of our functional requirements: the cloud video editor. This functional requirement, F_UC2_ 23 (see section 3.3.3) needs to manage and process the real time editing of the 360 videos. The 3D engine service needs to process and render different kinds of resolutions so they can be served with multiple levels of mesh details adapting to each device capability. The Translator service is the service that generates subtitles for audio and video automatically and delivers them in real time to the viewer.

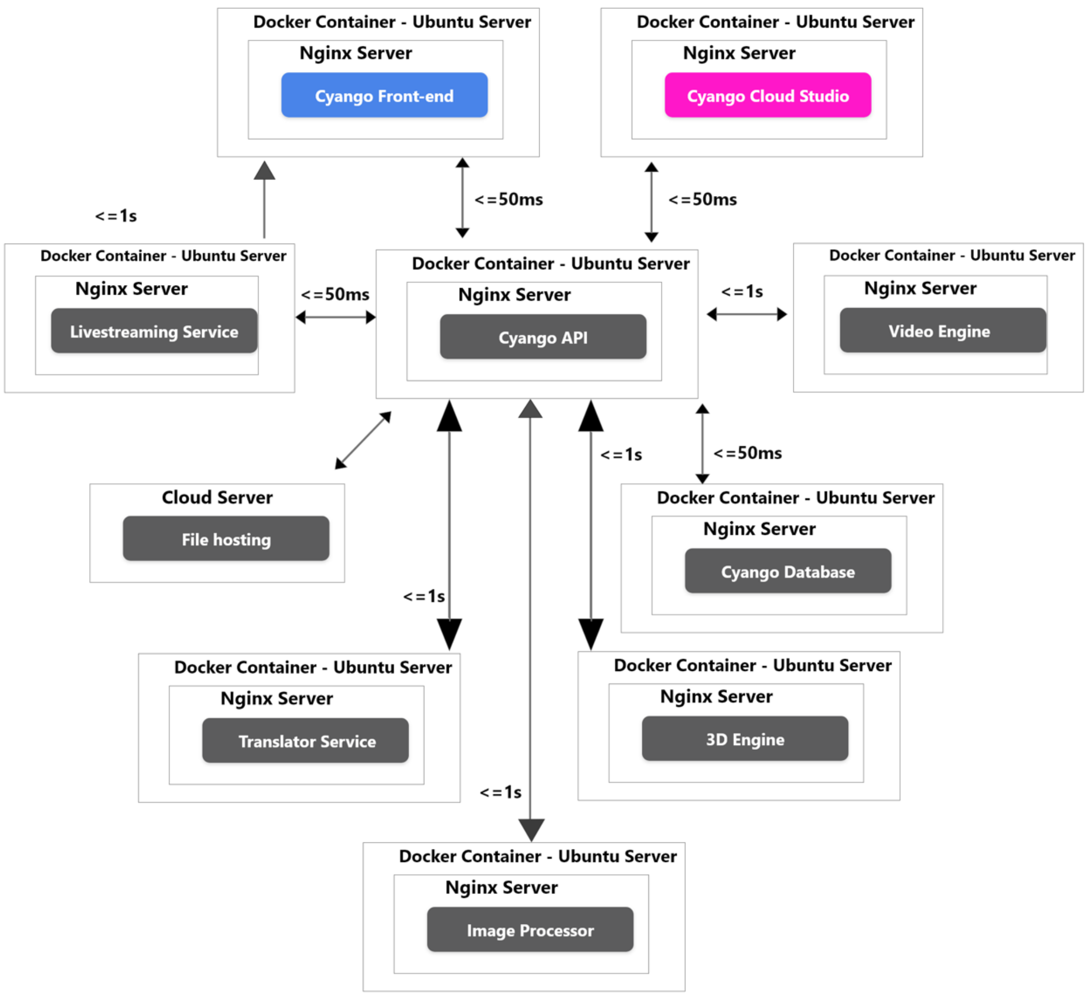


Figure 30: Future structural diagram of DOTES application.

2.2.2.2 Security/privacy aspects

To protect privacy and security of data, in the actual state of the platform, we are taking measures about the cookies management and the user registry. This information is encrypted. We have administrators that can delete information on the database directly upon user request. Sensitive information of user accounts is hashed using the SHA256 algorithm and traffic is encrypted using the AES-256 specification.

2.2.2.3 Reference scenarios

2.2.2.3.1 New Live Tour Scenario

A host wants to start a new live tour. He can start a live stream from his camera, e.g., using the RTMP protocol. Cyango must be able to have a powerful media server that can deliver the video with real time translations for text and for audio, so anyone in the world can watch it (see **Error! Reference source not found.**).

2.2.2.3.2 Streaming of high-quality 3D Model Scenario

A tour has a high-quality 3D Model. Cyango must be able to deliver the 3D model inside the tour generating and delivering different levels of detail of the 3D Model (see Figure 31).

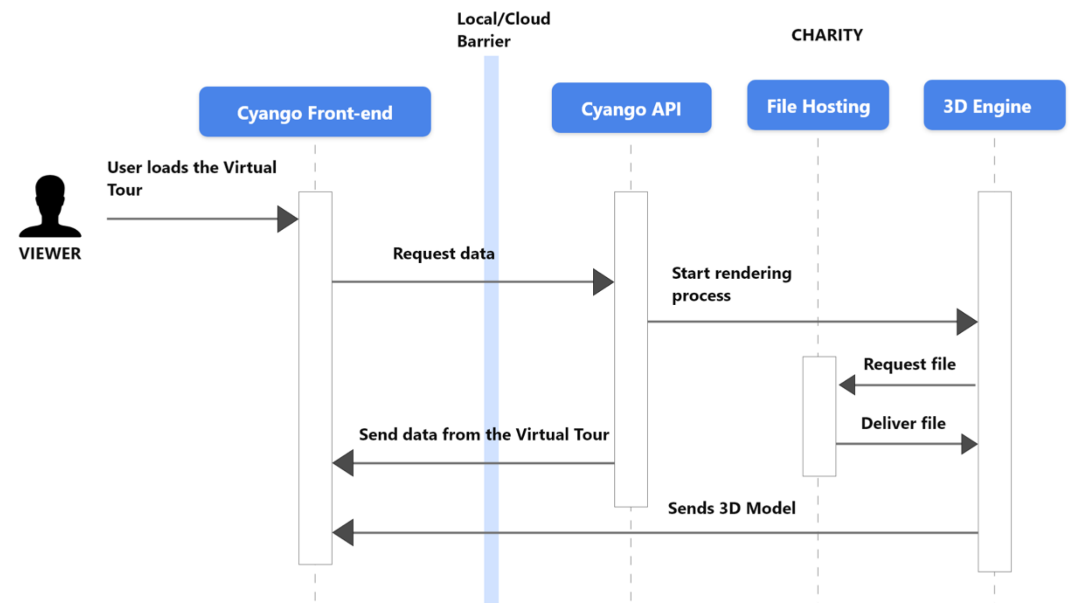


Figure 31: Streaming of high-quality 3D Model scenario.

2.2.2.3.3 Upload and download of a high-quality panorama Scenario

The user uploads a high-quality panoramic image. The cloud must be able to post-process it in tiles and deliver it (see Figure 32).

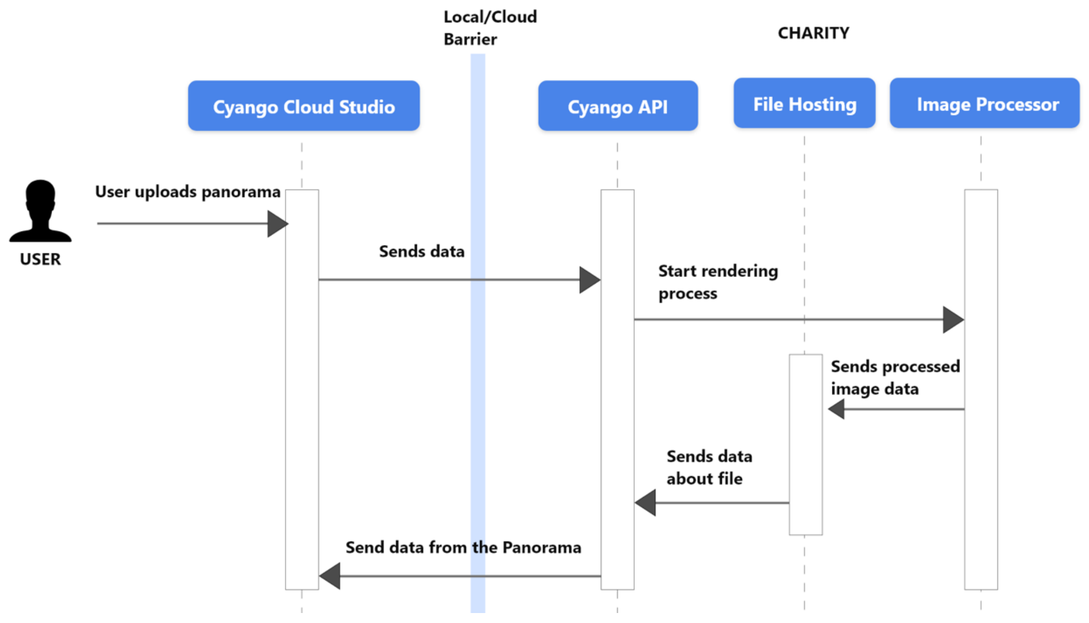


Figure 32: Upload and download of a high-quality panorama scenario.

2.2.2.3.4 Video edit through mobile network Scenario

After the user uploads the video through mobile network, he edits it and afterwards the front-end requests an action to the cloud service for processing and delivering the result of the action. This is related to the functional requirement F_UC2_22 (see 3.3.3) that improves quality of experience (QoE) (see Figure 33).

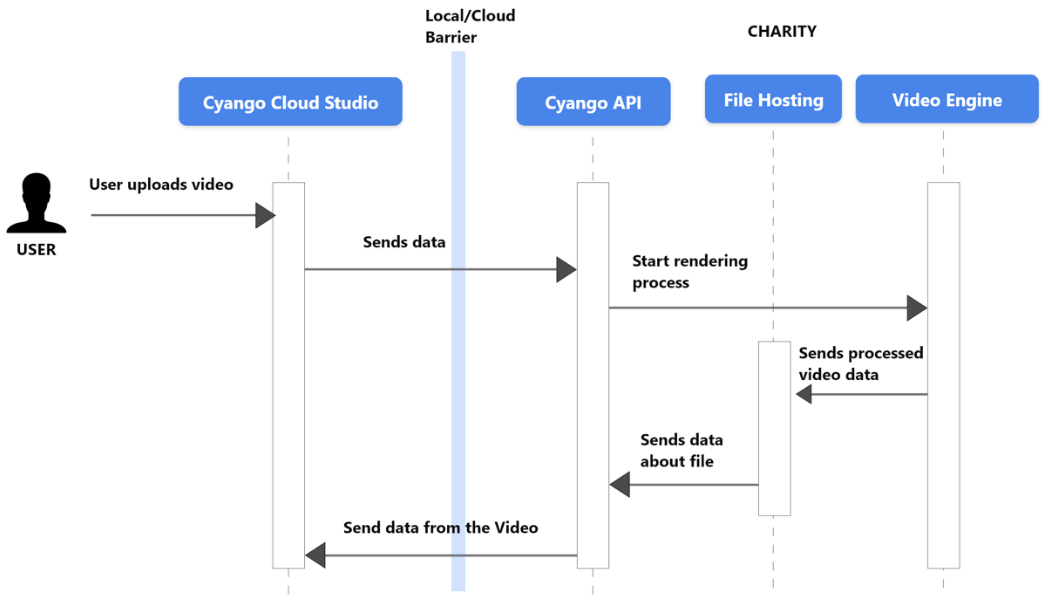


Figure 33: Video edit through mobile network scenario.

2.3 Use case 3 – Mixed Reality Interactive application

2.3.1 UC3-1 Collaborative Gaming Application

2.3.1.1 Overview

Orbital Knights (ORBK) will provide one of the Use Cases utilizing AR technology.

Main goal is to develop a highly immersive multiplayer AR game. To provide players with sufficient immersion, ORBK will develop a dedicated multiplayer engine which will be able to synchronize all dynamic game objects along with user's states throughout end devices. The overall solution we propose is based on the client-server architecture. The solution requires the infrastructure to provide key features: very low network latency and efficient resource discovery service, a trusted infrastructure (cloud/edge) to support Game Server from dishonest player's breaches.

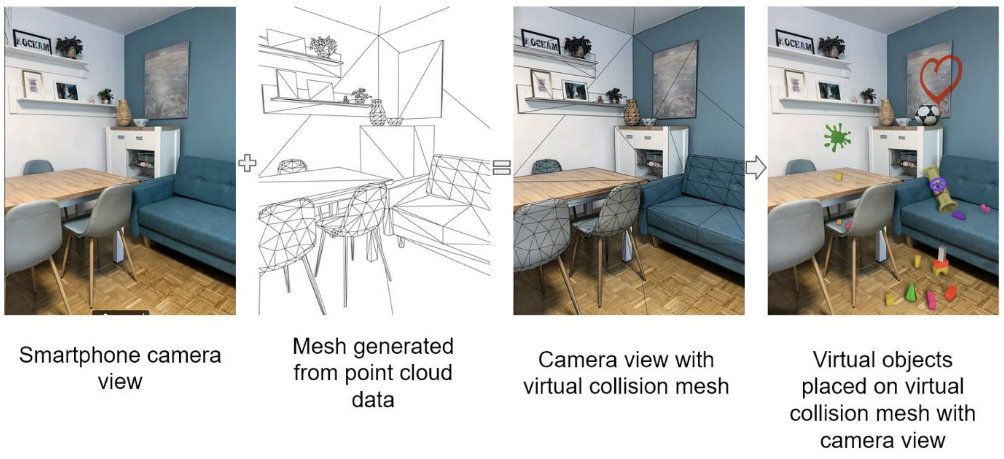

Smartphone camera view
Mesh generated from point cloud data
Camera view with virtual collision mesh
Virtual objects placed on virtual collision mesh with camera view

Figure 34: Mesh collider generation overview.

ORBK's ambition is to explore the potential of AR multiplayer games in the worldwide mobile games market. We are willing to research and identify the type and features of an AR multiplayer game that has the highest market potential.

We are also planning to use 3D Point Cloud technology to enrich gameplay and strengthen player's immersion, exploring CHARITY 3D Point Cloud support. It will use built-in device cameras' capabilities to provide input and use the output data to mix the real and virtual environments (see Figure 34).

2.3.1.2 Security/Privacy Aspects

Client application of the ORBK UC will scan real-life environment using device's built-in camera or sensors (like LiDAR). Scanning will result in gathering, temporary storing and processing 3D point cloud data to build mesh collider. Mesh collider will be used by game client and game server to simulate gameplay and synchronize/mix virtual and real environment. Both 3D point cloud data and mesh collider data will not be permanently stored (will be available only temporarily during the session) and will not be assigned to a particular user/person. Thus, ORBK UC will not pose any threats to privacy and security.

2.3.1.3 Reference scenarios

2.3.1.3.1 Submission of docker image with game server

Use Case provider enters CHARITY management website and provides credentials to log in. They can click a button to upload a docker image file. They can also see the history of 5 last uploaded docker images.

2.3.1.3.2 Player requesting a new game server

The User/Player starts the client application (game application) on their device. Then, the game client requests servers list API for list of nearby game servers. After receiving and displaying available game servers, player can decide to create a new game server. After clicking a button to start a new game server a request for a new game server is sent to CHARITY Deployment API. After receiving a request, CHARITY deploys a new game server on a hosting machine closest to the requesting player (based on geolocation). CHARITY Deployment API returns hosted game server IP to the client. Then, the client connects to hosted server though UDP protocol. (see Figure 35).

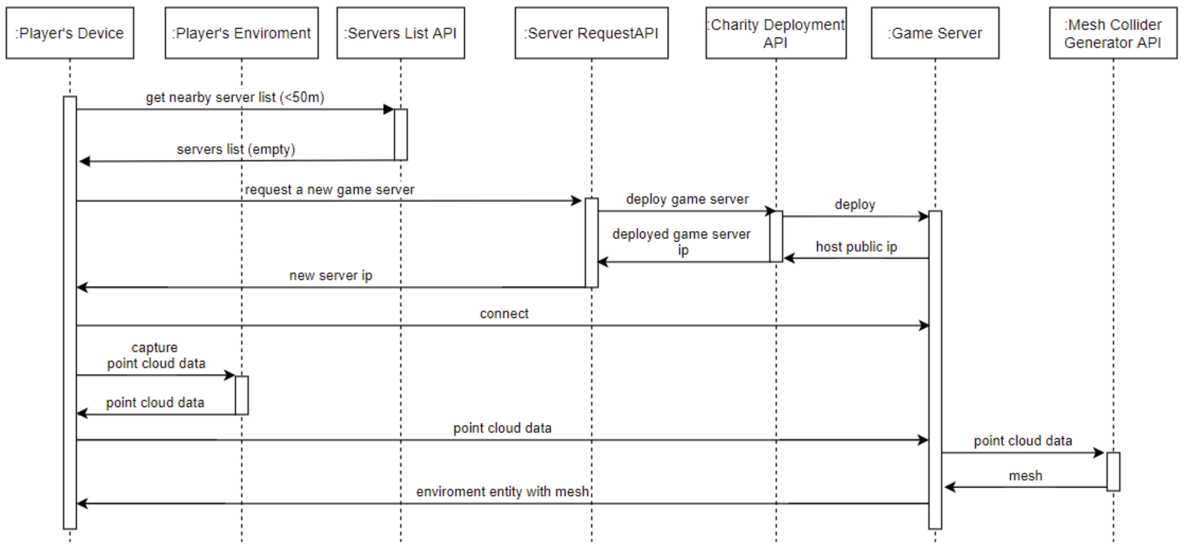


Figure 35: Player requesting a new game server.

2.3.1.3.3 Generating mesh collider from 3d point cloud data

After connecting to the game server, the player gets a notification to scan the environment to gather 3D point cloud data required for mesh collider generation. They move around with their phone pointing to the environment around. 3D point cloud API collects data during that process. When collecting is finished, the player clicks a button to send collected data to the game server. Game server sends that data further, to Mesh Collider Generator API. Mesh Collider Generator API generates a mesh collider based on received 3D point cloud data. Then it returns the generated mesh collider back to the game server. The game server sends a mesh collider to the client. (see Figure 35).

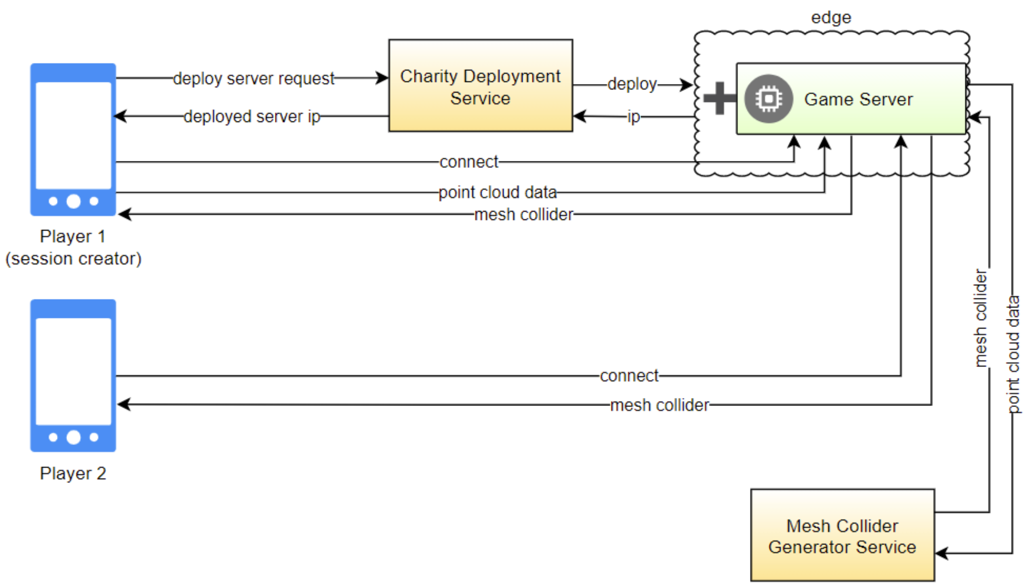


Figure 36: Player joining Game Server.

2.3.1.3.4 Player joining a game server

Player starts the client application (game application) on their device. Then, the game client requests servers list API for list of nearby game servers. Player clicks on one of the game servers on the list to join the existing game server. Client connects to selected game server though UDP protocol. After establishing a connection, game server sends a mesh collider to the connected client (see Figure 36).

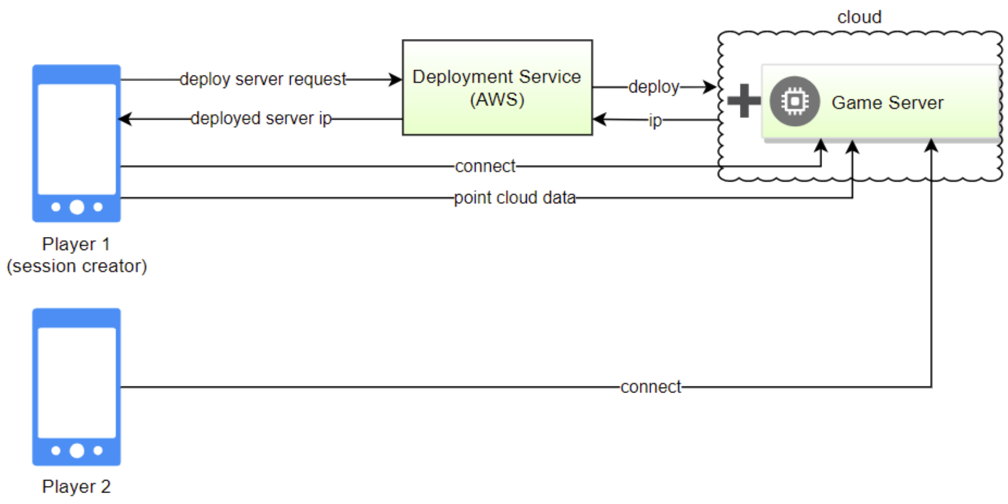


Figure 37: Current game server deployment pipeline.

Game server is currently deployed by hand to one fixed location at the AWS cloud network (Figure 37). CHARITY will allow us to deploy multiple game servers located as close as possible at one of the CHARITY resources. Automatic resource discovery and deployment will allow to shorten the distance between game server and game clients - in result lowering network latency and RTT.

CHARITY will also provide ORBK UC with mesh collider generation (based on gathered 3D point cloud data) and game client position discovery and synchronization services - both those high-quality services are necessary for the proper operation of the application (Figure 38).

CHARITY will allow this UC to build innovative AR gameplay and will contribute to improvement of overall user experience in AR games.

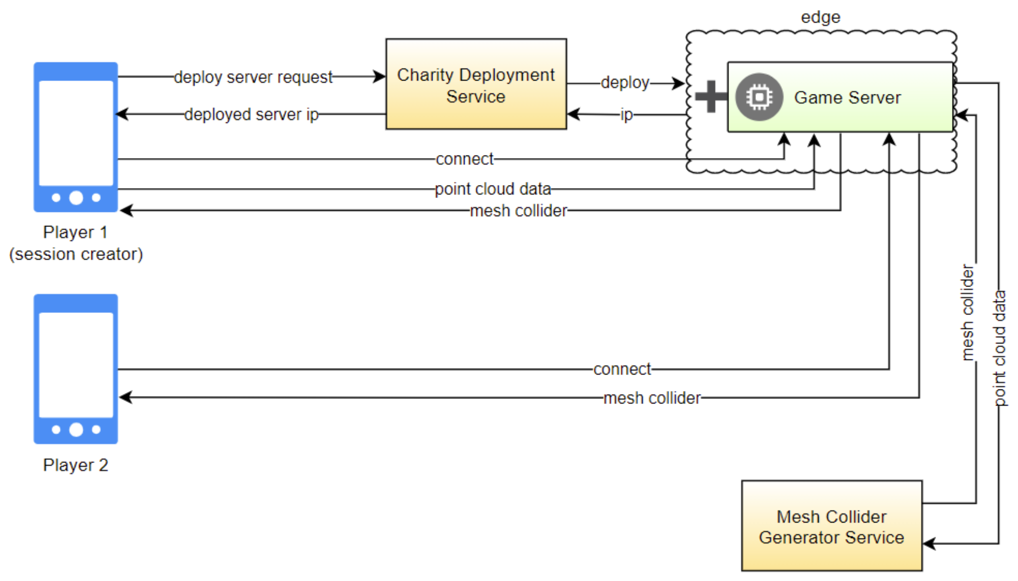


Figure 38: Expected CHARITY game server deployment pipeline.

2.3.2 UC3-2 Manned-Unmanned Operations Trainer Application

2.3.2.1 Overview

The Advanced Research & Technology (ART) operates as the applied research center for Collins Aerospace. Formerly known as UTRC, ART is seeking to leverage CHARITY advances for the Collins flight simulator product line. Flight simulators have a long history of extensive use in civilian aviation, playing an important role in reducing the overall cost of flight training. Simulators also allow potentially hazardous situations, such as aircraft failures and other emergencies to be safely experienced. Training simulators must accurately reflect the real world in order to serve as useful tools that can prepare trainees for real world challenges.

Trainees are immersed in a synthetically generated virtual world which is seamlessly integrated with a physical aircraft cockpit. As they look out from the window of the cockpit, they will see the virtual scenery generated by an image generator. However, as they look down to the cockpit, they will see their real controls and instruments along with their hands, feet, and anything else present in the cockpit. They can view and interact with their controls and instruments as needed to fly and navigate. In this way, they can build up the muscle memory related to critical flight skills.

The virtual scenery generated by the image generator is resource intensive. Background scenery will be rendered in the cloud and pre-fetched as required while foreground objects, such as other vehicles in close proximity, are rendered over the scenery in the edge. This collaborative rendering results in minimal rendering being executed in the real-time control path (see Figure 39) and minimizes round-trip time (RTT).

Trainees can virtually collaborate in a largely synthetic environment to perform coordinated search. They can also control remotely unmanned vehicles, such as simple aerial rescue drones, to approach inaccessible terrains in order to gain the situation awareness in the search-and-rescue scenario. In addition, all aircraft simulations need to be synchronized in the cloud and aircrafts in close proximity must be visible to each other on their screens.

Figure 39 illustrates the initial overview for this use case which is distributed over three layers: device, edge, and cloud. At the device layer, the trainee’s interaction with the cockpit is relayed to the Flight Dynamics component through Interaction Management. The Collins coreSIM component provides a complete flight dynamics simulation environment which calculates all the required flight related parameters and publishes a subset of these, including the aircraft position, altitude, and orientation in the form of a telemetry payload routed to the appropriate Edge node for updated scenery generation and as input to the Flight Oracle component.

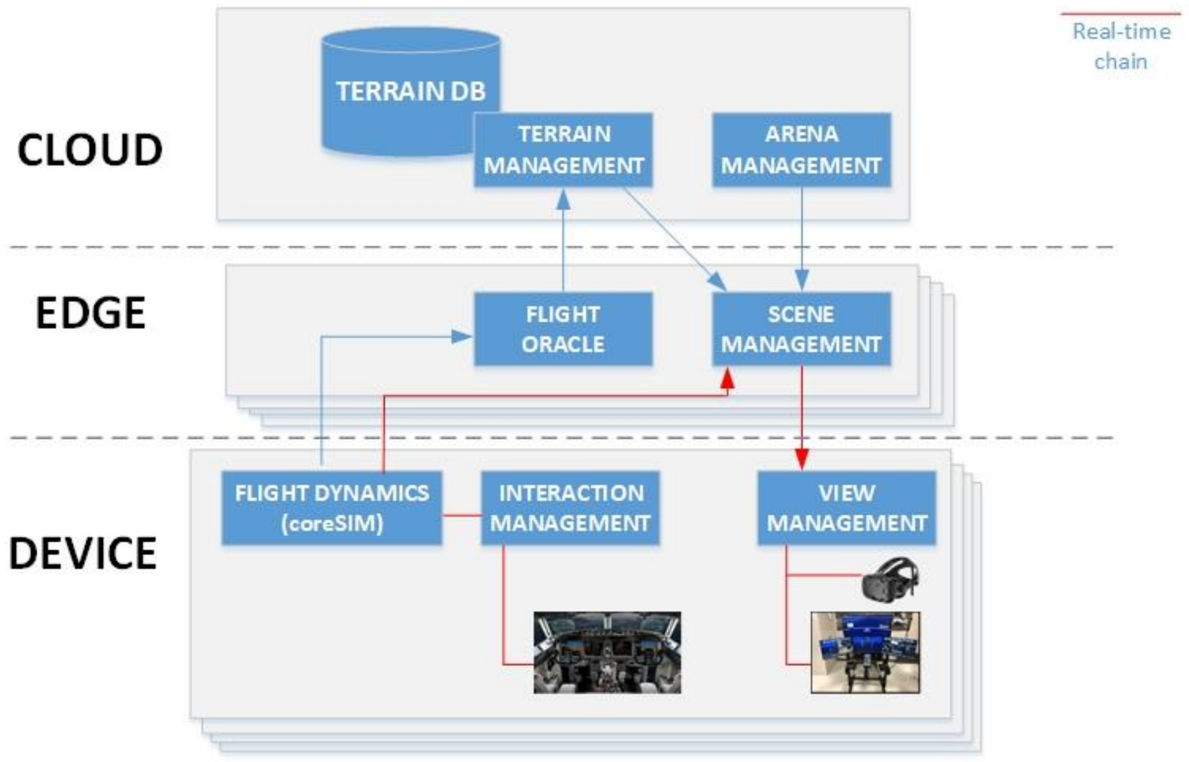


Figure 39: Initial overview of Training Simulator.

The Flight Oracle component predicts and pre-fetches rendered terrain imagery from the cloud to store in a local cache according to the registered flight path and observed trajectory during the course of the simulation. We seek to employ a scheme of collaborative rendering between the cloud and the edge in which the background terrain imagery will be rendered in the cloud and augmented with dynamic foreground elements at the edge.

Arena Management is hosted on the cloud to keep track of all aircraft currently deployed in a given group training simulation. This component maintains an up-to-date view of all aircraft positions and trajectories and relays this information as appropriate to relevant Scene Management components on the edge.

The objective is to restrict the latency-critical motion-to-photon chain to include only components hosted on the device and edge.

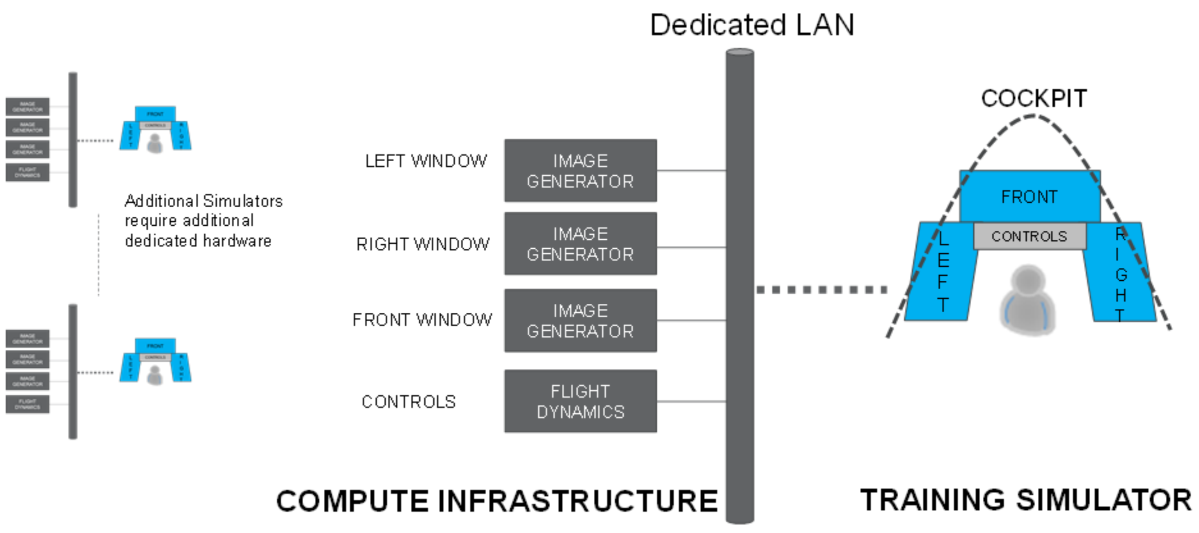


Figure 40: Current Model of Training Simulator Deployment.

The deployment of commercial flight simulators typically involves the shipping and installation of expensive equipment ranging from screens to cockpits supported by a local custom computing infrastructure that performs the necessary mission planning, simulation, rendering and user interaction mediation. If multiple training simulators are required on site, then they each have dedicated equipment. This is depicted in Figure 40.

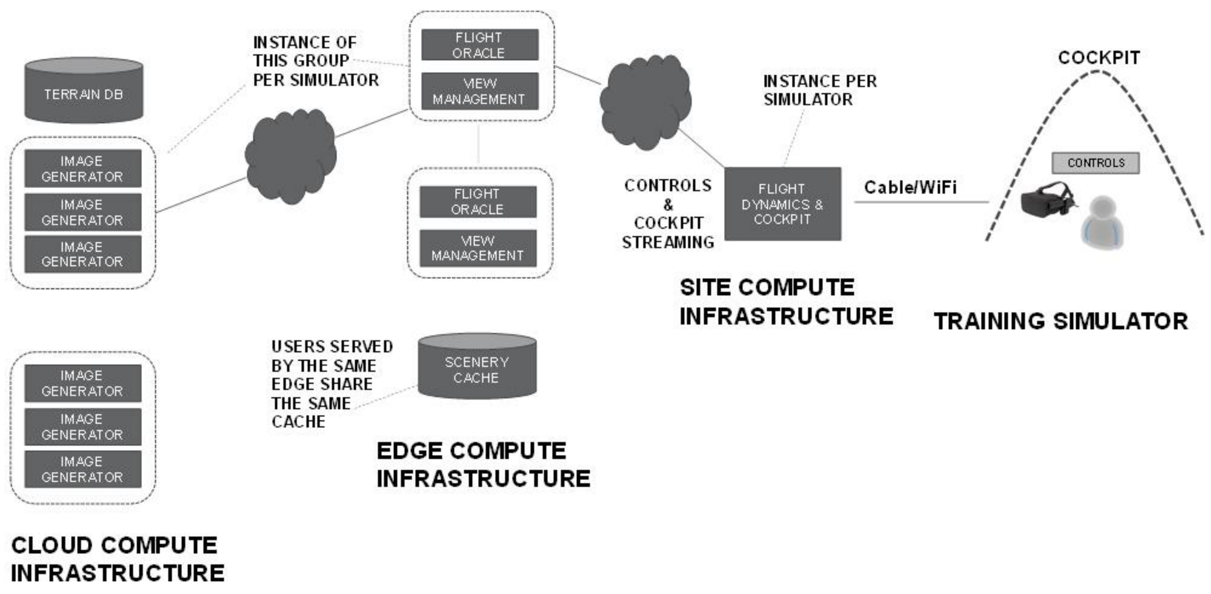


Figure 41: Target Architecture leveraging edge and cloud to scale.

ART seeks to advance the deployment of training simulators on the edge-cloud continuum and targets extended reality (XR) devices to deliver compelling immersive environments (Figure 41). Advances in containerization and orchestration are considered for enhancing software agility and mobility. In addition, we look to edge computing to engineer a new way forward that enables us to deliver high-end training simulation environments with minimal local hardware.

While simulation in the cloud is a key goal, we plan to leverage the fluidity of CHARITY deployment and orchestration to facilitate engagement and collaboration between multiple simulation instances. We will apply the CHARITY architecture to a collaborative distributed immersive training environment for emerging civil manned-unmanned teaming concepts.

2.3.2.2 Security/privacy aspects

Before using the flight simulator, a trainee has to login to secure resources across edge/cloud. In this way, a principal authentication would be needed. All activities of the trainee (e.g., movement and responses) during the training session are recorded against a unique, randomly generated, non-personally identifiable session ID which is stored in the database on the cloud data centre for operational purposes. Training participants in a joint simulation may be aware of other participant locations within the simulated terrain but not in a personally identifiable manner. This information is not shared with other third-party companies or applications. Other data which may be collected during the simulation include QoE and performance metrics along with fault and error reports. The data transferred from/to edge/cloud is completely anonymous and does not contain any personal data of the trainee.

2.3.2.3 Reference scenarios

2.3.2.3.1 Training Session Creation

Before conducting a training exercise, a trainee must login and establish a session. This serves the purpose of securing resources across the edge/cloud and establishing a communication path. All active users are registered with the Arena Manager on the cloud so that we can establish and maintain a centralised view of the status and position of all trainees. This is essential in the event that multiple trainees elect to join a combined exercise and share their airspace with other users.

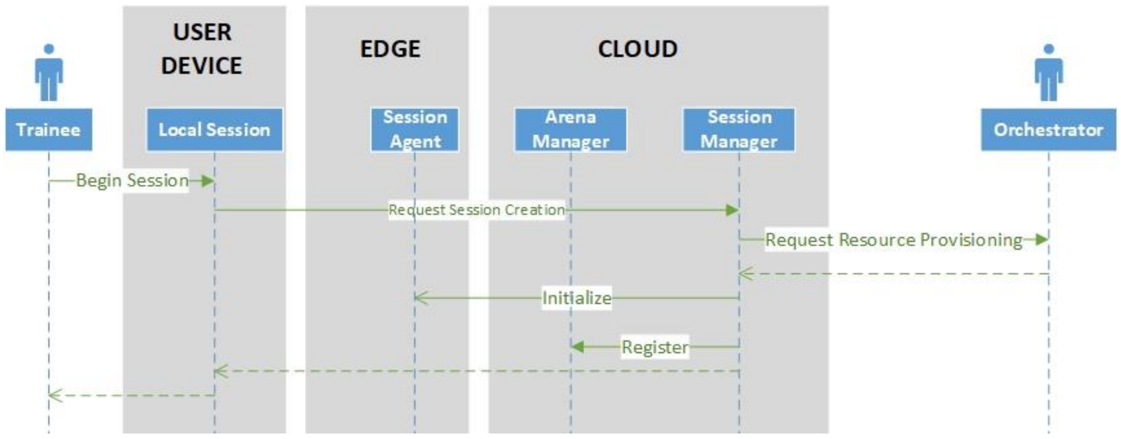


Figure 42: Sequence Diagram for establishing a training session.

Figure 42 shows the sequence diagram for establishing a training session. Session Agents are deployed on the edge node selected by the CHARITY Orchestrator and act as the sole point of contact for components running on the User Device. After the Local Session is created initially, it contains a handle on the associated Session Agent. Session Agents also serve as the point of contact for the Arena Manager in the event that it deems it necessary to inform a given simulator instance about other aircraft in its immediate vicinity.

2.3.2.3.2 Training Session Interaction

With a potentially large team of users (automated and human) taking part in a single training exercise, it is important that we have the means to optimize our use of bandwidth through reuse of assets and also the ability to aggressively cache content ahead of time, so the real-time co-ordination of a distributed team is not impacted by stragglers experiencing bandwidth or latency issues.

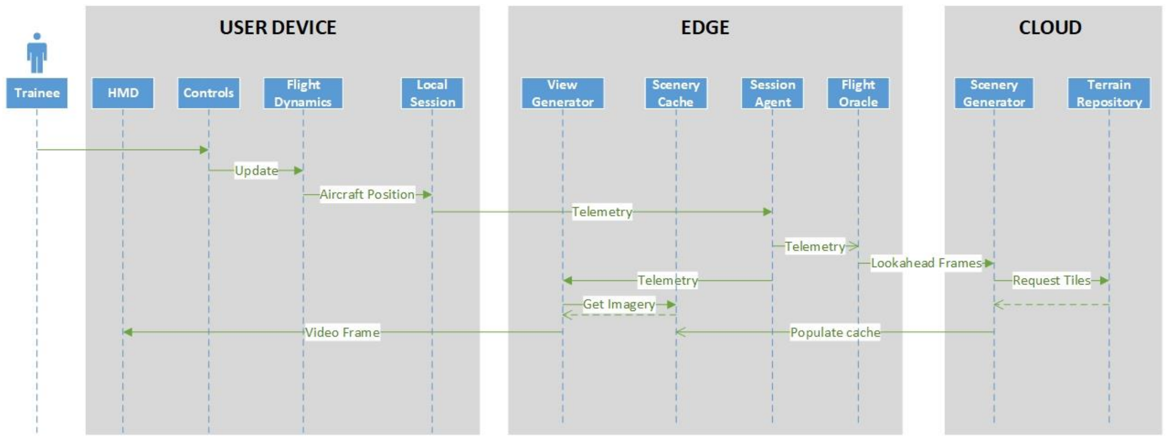


Figure 43: Sequence Diagram for user interacting with the simulator.

In Figure 43, the real-time path (so-called motion to photon) does not include the interaction with components in the cloud and depends on the necessary assets being in place in the Scenery Cache at the edge. The Flight Oracle is constantly monitoring and assessing the movement of the aircraft and ensuring that the cache is appropriately provisioned. If it observes an upcoming gap that would lead to a cache miss, then it contacts the scenery manager on the cloud to generate additional assets for the edge cache. Note that Cloud components would actually interact with the Session Agent on the edge, but we just depict direct communication with the Flight Oracle and Scenery Cache for the sake of clarity.

The Scenery Cache is shared amongst all trainees interacting with a given edge point which enables us to reuse imagery assets when multiple users operate in the same geographical area for a search and rescue operation.

2.3.2.3.3 Team Updates

The Arena Manager running on the cloud is responsible for keeping track of aircraft positions and notifying individual sessions of other aircraft in their immediate vicinity so that they can be reflected in the visual scenery presented to the end user (Figure 44).

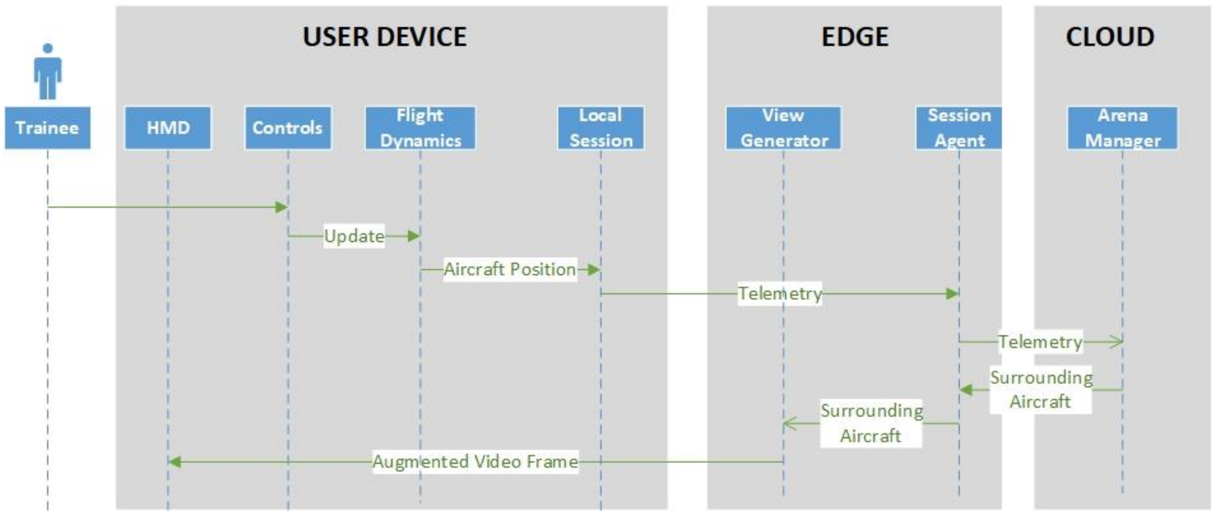


Figure 44: Sequence Diagram for Team Session Co-ordination.

3 CHARITY Requirements

3.1 Methodology

3.1.1 Introduction

The requirements presented in this document have been defined within the CHARITY consortium, considering several aspects such as use cases, performance requirements and architectural assumptions.

The CHARITY platform requirements were collected during many brainstorming sessions, internal consortium online meetings. The key aspect was to make them feasible in the first place, while leaving a space for technical research. In this process, technical challenges that form the use case specific requirements were taken into consideration. These are barely visible to the end user but affect QoE a great deal. The methodology involved gathering of all requirements into a single and lengthy list and, at a second stage, dividing them into 2 sub-groups to allow easier tracking and better readability. The sub-groups consist of:

- **General-purpose requirements:** Those requirements were provided by the use case owners, and they are key features of the XR applications that are common through all three use cases.
- **Use case related requirements:** Those requirements were extracted by analysing the needs of the XR applications. Since this analysis was conducted by the use case owners, they are filtered through several technical and business-related criteria.

Use case related requirements are further divided into functional and non-functional requirements. Functional requirements define specific behaviour, features, or functions. Non-functional requirements specify criteria and system properties that can be used to judge the operation of the system focusing on expectations from the system. A specific naming convention has been adopted and a set of requirements tables are provided in order to have a coherent description of the requirements throughout the document.

3.1.2 User Roles

This subsection describes the roles involved in the use case scenarios and creates a list of stakeholders' definitions to be used throughout the lifetime of the project. This is a generic description of the stakeholders involved in the CHARITY context along with the description of their responsibilities and benefits arising from their role.

ADMINISTRATOR: this role is responsible for operating and maintaining a secure, scalable, and efficient platform for the instantiation of CHARITY microservices. The responsibilities of CHARITY administrator involve edge pool management, proper and automated applications' deployment, providing data analytics regarding applications and edge resources. The Administrator is also responsible of analyzing and optimizing the resource allocation.

USER: all use case applications involve users as the consumers of a particular XR service i.e., Holographic concert musician, Holographic concert audience, Holographic meeting speaker, Holographic meeting audience, Holographic assistant, VR medical trainee, Virtual tour operator, Virtual tour viewer, AR gamer, Virtual flight trainee.

APPLICATION DEVELOPER: this role deals with use case partners developing innovative 5G-related XR services, leveraging on Virtual Network Functions (VNFs) and Network Function Virtualization (NFV) infrastructures to deploy, configure and manage their microservices. Developers aim at having a great variety of resources at their disposal with high quality of performance ensuring QoE for their USERS. The developers have the responsibility to develop, combine, integrate and customize functions (even developed by others) to create XR applications and microservices to be offered to USERS.

APPLICATION PROVIDER: this role deals with partners developing innovative 5G-related XR services hosted by CHARITY. In this respect, the role is responsible for the management of business aspects regarding the hosting of the application by CHARITY.

3.1.3 KPIs

This section describes the set of use case Key Performance Indicators (KPIs) that were originally defined within the CHARITY DoA. Each of them will be linked to the list of requirements presented in sections 3.2 and 3.3. The following table presents in three columns the defined KPIs:

- KPI ID: the ID of the requirement which will be used as reference in the list of requirements. Its naming convention is dependent to a particular use case category KPI-UC-X.YY where X is the UC category and YY is the id.
- Description: The column contains the textual description of the KPI.
- UC: The column lists the UC IDs that involve the specific KPI.

It should be noted that the following table lists two specific KPIs that were not originally defined in the CHARITY DoA, where a slightly different naming convention (i.e., KPI-UC-X.BYY) was used.

Table 2: Key Performance Indicators (KPIs)

KPI ID	Description	UC
KPI-UC-1.1	Average latency between sending input data and receiving 3D- point cloud <=60ms	UC1-1 UC1-2 UC1-3
KPI-UC-1.2	Decrease in bandwidth by 50%	UC1-1 UC1-2 UC1-3
KPI-UC-1.3	Frame rate of the holographic visualization >= 30Hz	UC1-1 UC1-2 UC1-3
KPI-UC-1.4	Data services required (raw data streaming, rendering, compression, caching, encoding) >=5	UC1-1 UC1-2 UC1-3
KPI-UC-1.5	(assistant) Latency in speech input (human) and speech output (avatar) <= 2 sec	UC1-3
KPI-UC-1.B1	(concerts & meetings) At most 5 video streams must be synchronized (not in GA)	UC1-1 UC1-2
KPI-UC-1.B2	(concerts & meetings) Holographic device fps 30-50 (not in GA)	UC1-1 UC1-2
KPI-UC-2.1	Average latency < 20 ms	UC2-1 UC2-2
KPI-UC-2.2	Number of CCUs in different geographic locations > 50	UC2-1 UC2-2
KPI-UC-2.3	Number of different VR HMDs >5	UC2-1 UC2-2
KPI-UC-2.4	Data services required (rendering, compression, caching, encoding) =>4	UC2-1 UC2-2
KPI-UC-2.5	Automated configurable soft-body simulation for objects with large number of vertices >= 8.000 vertices	UC2-1 UC2-2
KPI-UC-2.6	Server supporting up to 100 Virtual rooms	UC2-1

KPI ID	Description	UC
		UC2-2
KPI-UC-3.1	RTT (gaming) sum of network latency and game server response time < 100ms	UC3-1
KPI-UC-3.2	RTT (aeronautical) – sum of network latency and game server response time < 15ms	UC3-2
KPI-UC-3.3	Number of CCUs>30	UC3-1 UC3-2
KPI-UC-3.4	Number of synchronized AR objects >30	UC2-1 UC2-2
KPI-UC-3.5	Data services required (raw data streaming, rendering, compression, caching, encoding) >=5	UC2-1 UC2-2

3.1.4 Requirements' characteristics

Each requirement table in this document, except the general-purpose requirements, is defined by five columns:

- **ID**: the column containing the consolidated requirement set for CHARITY platform, clustered by an ID expressed by the following mentioned naming convention. This signature divided in three parts: <reqType>_<reqCategory>_<reqID>
  - <reqType>: identifies the type of requirement replaced with one of the following:
    - F: meaning Functional requirement",
    - NF: meaning "Non-functional requirement ",
  - <reqCategory>: identifies the category of the requirement:
    - GP: General purpose
    - UCx: Use case #x (x could be 1, 2, or 3)
  - <reqID>: is the number that identifies the requirement.
- **Description**: a textual description of the requirement. It may also provide the user role that is responsible for fulfilling the requirement or major system component.
- **Req. Level**: indicates the level of priority for the requirement:
  - MUST: Must be met at all cost
  - SHOULD: recommended to be met
  - MAY: Optional
- **Reporter**: indicates the consortium member reporting the specific requirement
- **KPI**: identifies the Key Performance Indicators (KPIs) that is linked to the specific requirement

3.2 General purpose requirements

Below is a list of general requirements for the CHARITY platform. These requirements are mainly related to the platform and user management.

Table 3: General-purpose requirements

ID	Description	Req. Level	KPI
NF_GP_01	The CHARITY infrastructure must be able to provide	MUST	KPI-UC-1.4

ID	Description	Req. Level	KPI
	authentication for user access to deployed pipelines.		KPI-UC-2.4 KPI-UC-3.5
NF_GP_02	The CHARITY infrastructure must monitor the responsiveness of components.	MUST	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
NF_GP_03	The CHARITY infrastructure may require monitoring functions to be included in components for responsiveness monitoring (e.g., watchdog support).	MAY	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
F_GP_04	The CHARITY infrastructure must log all actions taken on hosted components.	MUST	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
F_GP_05	The CHARITY infrastructure should provide support for logging status & quality of service (QoS) metrics from hosted components.	SHOULD	KPI-UC-1.1 KPI-UC-2.1 KPI-UC-3.5
NF_GP_06	The CHARITY infrastructure must provide the ability to troubleshoot a distributed pipeline (without having to log into a collection of disparate machines to examine logs).	MUST	KPI-UC-3.5
F_GP_07	The CHARITY infrastructure must support components hosted within the core/cloud, edge, and on user premises.	MUST	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
F_GP_08	The CHARITY orchestrator must support runtime migration of hosted components between physical hosting locations.	MUST	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-3.5
NF_GP_09	The CHARITY orchestrator must support multiple component failure management policies, including at a minimum component restart, and failover/hotswap.	MUST	KPI-UC-3.5
NF_GP_10	The CHARITY orchestrator must support duplicated and synchronized hosted components (e.g., as a means of latency control).	MUST	KPI-UC-1.1 KPI-UC-2.1 KPI-UC-3.2
NF_GP_11	The CHARITY orchestrator should support the ability to insert new components within an operating pipeline, without loss of data.	SHOULD	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
NF_GP_12	The CHARITY orchestrator must support construction of a processing pipeline from a directed acyclic graph (DAG) specification.	MUST	KPI-UC-3.5
NF_GP_13	The CHARITY orchestrator must support multiple concurrent pipelines with centralized cloud co-ordination.	MUST	KPI-UC-2.2 KPI-UC-3.3
NF_GP_14	The CHARITY orchestrator should be able to validate data and service dependencies between components in the pipeline.	SHOULD	KPI-UC-1.4 KPI-UC-2.4 KPI-UC-3.5
NF_GP_15	The CHARITY orchestrator must be able to support arbitrary multimedia data streams between components in the pipeline.	MUST	KPI-UC-3.5
NF_GP_16	Automatic restart of unresponsive components may be accomplished by restarting the component (restart) or by	MAY	KPI-UC-1.4 KPI-UC-2.4

ID	Description	Req. Level	KPI
	swapping to a redundant component (failover).		KPI-UC-3.5

3.3 Use case specific requirements

In this section, we present the functional and non-functional requirements for each use case.

3.3.1 Use case category 1 – Functional requirements

Table 4: UC category 1 - Functional requirements

ID	Description	Req. Level	Reporter	KPI
F_UC1_01	CHARITY provides Cloud server with resources necessary to achieve KPIs.	MUST	HOLO3D	KPI-UC-1.B2 KPI-UC-1-1 KPI-UC-1.2 KPI-UC-1.3 KPI-UC-1-4
F_UC1_02	CHARITY provides cloud-based software to receive, decompress and render / modify the content in the cloud in real time.	MUST	HOLO3D	KPI-UC-1.B2 KPI-UC-1-1 KPI-UC-1.2 KPI-UC-1.3 KPI-UC-1-4
F_UC1_03	CHARITY software renders in real time several types of pre-set video modes and resolutions, for several types of Holographic Displays.	SHOULD	HOLO3D	KPI-UC-1.B2 KPI-UC-1-1 KPI-UC-1.2 KPI-UC-1.3 KPI-UC-1-4
F_UC1_04	APPLICATION PROVIDER provides speaker PC, video camera, lights, black background, secondary screen.	MAY	HOLO3D	KPI-UC-1-1
F_UC1_05	APPLICATION PROVIDER provides speaker PC with software to retrieve the raw, 2D video from the video camera and send it to the Cloud server.	MUST	HOLO3D	KPI-UC-1-1
F_UC1_06	APPLICATION PROVIDER provides client PC, Holographic Display, webcam, mic for 2-way communication with the Speaker PC.	MUST	HOLO3D	KPI-UC-1-1
F_UC1_07	APPLICATION PROVIDER provides client PC with software to send live video/sound stream to the Cloud server.	MUST	HOLO3D	KPI-UC-1-1
F_UC1_08	APPLICATION PROVIDER provides client PC with software to receive the scrambled, 3D adapted video from the Cloud Server and send it to the Holographic Display.	MUST	HOLO3D	KPI-UC-1.B2 KPI-UC-1.3
F_UC1_09	APPLICATION PROVIDER provides client PC with software to synchronize with the other connected client PCs.	MUST	HOLO3D	KPI-UC-1.B1
F_UC1_10	APPLICATION PROVIDER, the software F_UC2_08 can choose to retrieve a different type of scrambled, 3D adapted stream from the Cloud	SHOULD	HOLO3D	KPI-UC-1-4

ID	Description	Req. Level	Reporter	KPI
	server according to the connected Holographic Display.			
F_UC1_11	APPLICATION PROVIDER provides speaker PC with software to convert the shared content (jpg, pdf, doc, ppt, mp4) to the same type of raw,2d video as in F_UC1_05 (Holographic meetings scenario).	MUST	HOLO3D	KPI-UC-1-1
F_UC1_12	CHARITY provides Windows Server VM with powerful Nvidia GPU for Unity 3D Rendering and GPU post processing preferably on edge cloud.	MUST	SRT	KPI-UC-1.1
F_UC1_13	APPLICATION PROVIDER provides Unity software to render the content and assistant.	MUST	SRT	KPI-UC-1.1
F_UC1_14	APPLICATION PROVIDER receives eye position data and interaction data from client device.	MUST	SRT	
F_UC1_15	APPLICATION PROVIDER provides software to generate the 3D point cloud from rendered content frame by frame.	MUST	SRT	KPI-UC-1.1
F_UC1_16	APPLICATION PROVIDER generates additional occlusion data.	SHOULD	SRT	KPI-UC-1.1
F_UC1_17	APPLICATION PROVIDER provides software to model behavior of the holographic assistant.	MUST	SRT	KPI-UC-1.5
F_UC1_18	APPLICATION PROVIDER uses cloud services to process speech – i.e., google cloud text to speech and speech to text.	MUST	SRT	KPI-UC-1.5
F_UC1_19	APPLICATION PROVIDER uses cloud services to gather information – weather, stocks dependent upon requests from USER.	MUST	SRT	
F_UC1_20	APPLICATION PROVIDER uses cloud services to connect to a chat bot for smooth ping pong interactions – i.e., from IBM Watson.	SHOULD	SRT	
F_UC1_21	APPLICATION PROVIDER is able to visualize information.	MUST	SRT	KPI-UC-1.1
F_UC1_22	APPLICATION PROVIDER uses artificial intelligence to interpret speech (=text from speech to text) of USER to execute appropriate requests.	SHOULD	SRT	
F_UC1_23	APPLICATION PROVIDER receives speech data from client device.	MUST	SRT	KPI-UC-1.5
F_UC1_24	APPLICATION PROVIDER uses other cloud services to gather information.	MAY	SRT	
F_UC1_25	CHARITY provides Windows Server VM with powerful CPU (optional GPU) on edge cloud for point cloud processing.	MUST	SRT	KPI-UC-1.1
F_UC1_26	CHARITY provides a software to encode 3D point cloud data (optional using GPU acceleration) into a compressed format capable for network transport and streaming and sends it to client device.	MUST	SRT	KPI-UC-1.1 KPI-UC-1.2 KPI-UC-1.4
F_UC1_27	CHARITY optimizes bandwidth by only transferring differences between 3D point cloud frames.	SHOULD	SRT	KPI-UC-1.1 KPI-UC-1.2
F_UC1_28	CHARITY provides a software running on a client device to decode 3D point cloud data from network data stream.	MUST	SRT	KPI-UC-1.1 KPI-UC-1.2
F_UC1_29	APPLICATION PROVIDER provides a software running on a client device to compute received point cloud data dependent from local eye-tracking data and optionally provided occlusion data into holograms.	MUST	SRT	KPI-UC-1.3
F_UC1_30	APPLICATION PROVIDER records USERs voice and	MUST	SRT	KPI-UC-1.5

ID	Description	Req. Level	Reporter	KPI
	outputs assistant speech.			

3.3.2 Use case category 1 – non-Functional requirements

Table 5: UC category 1 - non-Functional requirements

ID	Description	Req. Level	Reporter	KPI
NF_UC1_01	The video resolution should be > than full HD (1920x1080) @ 30 fps.	SHOULD	HOLO3D	KPI-UC-1.3
NF_UC1_02	Average latency between receiving the raw, 2D video stream from Speaker PC and rendering it for the specific Holo Display resolution and format required by the Client PC<= 30-600 Seconds.	SHOULD	HOLO3D	KPI-UC-1.B2
NF_UC1_03	Average latency between receiving the raw, 2D video stream from Speaker PC and rendering it for the specific Holographic Display resolution and format required by the Client PC<=1000ms (second scenario).	SHOULD	HOLO3D	KPI-UC1-1.B2
NF_UC1_04	Average latency between sending input data and receiving 3D- point cloud <=60ms.	MUST	SRT	KPI-UC-1.1
NF_UC1_05	Data services required (raw data streaming, rendering, compression, caching, encoding) >=5.	MUST	SRT	KPI-UC-1.4
NF_UC1_06	Latency in speech input (human) and speech output (assistant) <= 2 sec.	MUST	SRT	KPI-UC-1.5
NF_UC1_07	Network bandwidth between edge servers and client device is appropriate to achieve KPIs.	MUST	SRT	KPI-UC-1.1
NF_UC1_08	For each client an instance of required VMs is provided via orchestration.	MUST	SRT	
NF_UC1_09	There is a mechanism to easily provide new versions of the VM images / update software in the VM images.	MUST	SRT	

3.3.3 Use case category 2 – Functional requirements

Table 6: UC Category 2 Functional requirements

ID	Description	Req. Level	Reporter	KPI
F_UC2_01	APPLICATION DEVELOPER: Use the mirror networking service or similar for matchmaking, creation of session and selection of an already existing session (IP, location, userid master) photon.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.6
F_UC2_02	USER: Able to create session and/or select an existing session from the application on the HMD based on credentials.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.6
F_UC2_03	APPLICATION DEVELOPER: Session management through a relay server or message broker in the cloud.	SHOULD	ORAMA	KPI-UC-2.2 KPI-UC-2.6
F_UC2_04	USER: Application updates through cloud-based repository, including apk installation on HMD.	MUST	ORAMA	KPI-UC-2.3
F_UC2_05	APPLICATION DEVELOPER: Communication of the application with Azure Cloud to store and retrieve user analytics.	MUST	ORAMA	KPI-UC-2.1
F_UC2_06	USER: Able to visualize performance analytics on	SHOULD	ORAMA	KPI-UC-2.1

ID	Description	Req. Level	Reporter	KPI
	the HMD.			
F_UC2_07	APPLICATION DEVELOPER: The application component running on the HMD should be aware of the connected resources (app instance on edge) where part of the application has been offloaded.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3 KPI-UC-2.4
F_UC2_08	APPLICATION DEVELOPER: The application running on the HMD should be able to connect via standardized protocols to the resources (app instance on edge) where part of the application has been offloaded.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3
F_UC2_09	APPLICATION DEVELOPER: Able to send output from the controllers and HMD to the resource (app instance on edge) where part of the application has been offloaded.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3
F_UC2_10	APPLICATION DEVELOPER: Able to receive encoded image data to the HMD for display from the resource (app instance on edge) where part of the application has been offloaded.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.4 KPI-UC-2.5
F_UC2_11	APPLICATION DEVELOPER: Support continuous streaming of two images (one per eye) per user from the edge resource node to the HMD.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.4 KPI-UC-2.5
F_UC2_12	APPLICATION DEVELOPER: Application component on the untethered HMD needs to be deployed and run on ARM based architecture (to support SoC).	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3
F_UC2_13	APPLICATION DEVELOPER: The resource discovery mechanism of CHARITY should offload part of the application functionality from the HMD to nearby edge resource considering lowest average latency.	MUST	ORAMA	KPI-UC-2.1
F_UC2_14	APPLICATION DEVELOPER: Efficient sharing of the same resource among different users.	SHOULD	ORAMA	KPI-UC-2.1 KPI-UC-2.6
F_UC2_15	APPLICATION DEVELOPER: The send rate of rotations/translations from the HMD should be dynamically adapted considering the network characteristics.	SHOULD	ORAMA	KPI-UC-2.1
F_UC2_16	USER, APPLICATION DEVELOPER: Able to start the application from the HMD without any additional system configuration.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3
F_UC2_17	APPLICATION DEVELOPER: Establish communication of the HMD and the Remote Service (RS) in the cloud/edge when launching the app on the HMD.	MUST	ORAMA	KPI-UC-2.1
F_UC2_18	APPLICATION DEVELOPER: Establish communication and initial data transfer from edge node for the user to enter the Virtual Operating Room.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.3 KPI-UC-2.6
F_UC2_19	APPLICATION DEVELOPER: scene, data assets and avatars are locally stored on the edge node.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.5
F_UC2_20	APPLICATION DEVELOPER: Different instances of the application can run on the same edge node supporting different users via their HMD devices.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.3
F_UC2_21	APPLICATION DEVELOPER: The HMD should be able to receive data from different modules running on different edge nodes.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2
F_UC2_22	APPLICATION DEVELOPER: Cloud video editor. Allows USER to edit the video file on the cloud with tools like trim, transitions and color grading.	MUST	DOTES	KPI-UC-2.1 KPI-UC-2.4

ID	Description	Req. Level	Reporter	KPI
F_UC2_ 23	APPLICATION DEVELOPER: Real-time video streaming. The VIEWER must be able to consume the live streaming video on the Story Front-end.	SHOULD	DOTES	KPI-UC-2.1
F_UC2_24	APPLICATION DEVELOPER Real-time 3D Model server-side render. The VIEWER should be able to see the 3D model with adaptative quality depending on the network quality.	SHOULD	DOTES	KPI-UC-2.1 KPI-UC-2.4
F_UC2_25	APPLICATION DEVELOPER: Real-time audio translation. The edge cloud should be able to process the audio of a live streaming video and transcribe it on the APPLICATION DEVELOPER: Cloud processing power. The edge cloud should have enough resource allocation depending on the demand of the media files that the VIEWER requests.	SHOULD	DOTES	KPI-UC-2.1
F_UC2_26	APPLICATION DEVELOPER: Cloud video editor. Allows USER to edit the video file on the cloud with tools like trim, transitions and color grading.	MUST	DOTES	KPI-UC-2.4

3.3.4 Use case category 2 – non-Functional requirements

Table 7: UC category 2 - non-Functional requirements

ID	Description	Req. Level	Reporter	KPI
NF_UC2_01	USER, APPLICATION DEVELOPER: Round trip time (RTT) latency <15ms.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_02	USER, APPLICATION DEVELOPER: Data rate >50 Mbps supported by at least 5Ghz wifi or 5G.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_03	USER, APPLICATION DEVELOPER: Minimum connectivity requirements of 5GHz wifi or 5G.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_04	USER, APPLICATION DEVELOPER: Connectivity from user HMD device <10 ms.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_05	USER: HMD able to support minimum frame rate of 60 fps (depending on the HMD).	MUST	ORAMA	KPI-UC-2.1
NF_UC2_06	USER: HMD supporting Full HD with minimum refresh rate 60Hz.	MUST	ORAMA	KPI-UC-2.3
NF_UC2_07	ADMINISTRATOR: Support at least 50 CCUs in the same Operating Room.	MUST	ORAMA	KPI-UC-2.2
NF_UC2_08	ADMINISTRATOR: The CHARITY framework to ensure security of the software components on the end devices and computing resources and transferred data across the network.	MUST	ORAMA	
NF_UC2_09	ADMINISTRATOR, APPLICATION DEVELOPER: To be able to use existing networks and available infrastructures.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_10	APPLICATION DEVELOPER: Receive error messages on potential problems with existing resources, continue the VR app by communicating with another newly discovered resource (discovery and placement).	MUST	ORAMA	KPI-UC-2.1
NF_UC2_11	USER: Continue using the application in case of problems in network resources with minimal delay.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_12	ADMINISTRATOR: Maintain application integrity and user’s security.	MUST	ORAMA	

ID	Description	Req. Level	Reporter	KPI
NF_UC2_13	APPLICATION DEVELOPER, USER: Proximity of the relay server based on the users' footprints.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2
NF_UC2_14	APPLICATION DEVELOPER: GPU and CUDA acceleration capabilities available at edge nodes, where part of the application is instantiated.	MUST	ORAMA	KPI-UC-2.4
NF_UC2_15	USER, ADMINISTRATOR: Support all current and upcoming untethered HMDs with different resolutions.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.3
NF_UC2_16	USER: Support action and haptic feedback in real time.	MUST	ORAMA	KPI-UC-2.1
NF_UC2_17	USER, APPLICATION DEVELOPER: Performed actions from all users must be synchronized to the output rendered image of each individual user's HMD with lowest average latency.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.4 KPI-UC-2.5
NF_UC2_18	USER: Virtual tools exchange must be flawless between the participants in the same Operating Room.	MUST	ORAMA	KPI-UC-2.1 KPI-UC-2.2 KPI-UC-2.4 KPI-UC-2.5
NF_UC2_19	USER, APPLICATION DEVELOPER: Data rate >50 Mbps supported by at least 5Ghz wifi or 5G.	MUST	DOTES	KPI-UC-2.1
NF_UC2_20	APPLICATION DEVELOPER: Receive error messages on potential problems with existing resources, continue the VR app by communicating with another newly discovered resource (discovery and placement).	MUST	DOTES	KPI-UC-2.1
NF_UC2_21	USER: Continue using the application in case of problems in network resources with minimal delay.	MUST	DOTES	KPI-UC-2.1
NF_UC2_22	ADMINISTRATOR: Maintain application integrity and user's security.	MUST	DOTES	
NF_UC2_23	APPLICATION DEVELOPER, USER: Proximity of the relay server based on the users' footprints.	MUST	DOTES	KPI-UC-2.1 KPI-UC-2.2
NF_UC2_24	APPLICATION DEVELOPER: GPU and CUDA acceleration capabilities available at edge nodes, where part of the application is instantiated.	MUST	DOTES	KPI-UC-2.4

3.3.5 Use case category 3 – Functional requirements

Table 8: UC category 3 – Functional requirements

ID	Description	Req. Level	Reporter	KPI
F_UC3_1	ADMINISTRATOR: CHARITY should have a repository which will store docker images.	SHOULD	ORBK	KPI-UC-3.5
F_UC3_2	ADMINISTRATOR: CHARITY should have a website (CHARITY management website) which can be used to update docker images to docker images repository.	SHOULD	ORBK	KPI-UC-3.5
F_UC3_3	ADMINISTRATOR: CHARITY management website should display docker images uploaded to docker images repository.	SHOULD	ORBK	KPI-UC-3.5
F_UC3_4	ADMINISTRATOR: CHARITY management website should display deployed docker images status.	SHOULD	ORBK	KPI-UC-3.5
F_UC3_5	APPLICATION DEVELOPER: CHARITY must have a deployment API which can be used to request for a new game server instance.	MUST	ORBK	KPI-UC-3.5

ID	Description	Req. Level	Reporter	KPI
F_UC3_6	APPLICATION DEVELOPER: CHARITY must be able to deploy docker images.	MUST	ORBK	KPI-UC-3.5
F_UC3_7	APPLICATION DEVELOPER: Deployment API must return host public IP after deploying docker image.	MUST	ORBK	KPI-UC-3.5
F_UC3_8	APPLICATION DEVELOPER: Deployed docker image must be reachable by UDP protocol through one of predefined ports.	MUST	ORBK	KPI-UC-3.5
F_UC3_9	APPLICATION DEVELOPER: Deployment API must receive GPU, CPU and RAM requirements when requesting a new game server.	MUST	ORBK	KPI-UC-3.5
F_UC3_10	APPLICATION DEVELOPER: CHARITY must have a service which will generate mesh collider from 3d point cloud.	MUST	ORBK	KPI-UC-3.5
F_UC3_11	APPLICATION DEVELOPER: CHARITY must deploy docker image as close (geolocation) to requesting player as possible (with lowest latency).	MUST	ORBK	KPI-UC-3.5
F_UC3_12	ADMINISTRATOR: CHARITY should monitor deployed docker image status (CPU usage, RAM usage, overall performance).	SHOULD	ORBK	KPI-UC-3.5
F_UC3_13	The simulation must facilitate collaboration between users to efficiently execute the simulated mission.	MUST	UTRC	KPI-UC-3.3 KPI-UC-3.5
F_UC3_14	Scenery generation may support scenery with different weather.	MAY	UTRC	KPI-UC-3.5
F_UC3_15	The simulated environment should allow participants to join or leave simulation at any time.	SHOULD	UTRC	KPI-UC-3.5
F_UC3_16	The simulation should enable prediction of background scenery demands so that it can be pre-fetched by any component from off-line storage.	SHOULD	UTRC	KPI-UC-3.5
F_UC3_17	The simulation should enable custom tiling of cloud-based image generator output to facilitate variable resolution across a single frame.	SHOULD	UTRC	KPI-UC-3.5
F_UC3_18	The simulation should be able support both active participants (present in the simulated environment) and passive observers (not present in the simulate environment).	SHOULD	UTRC	KPI-UC-3.3

3.3.6 Use case category 3 – non-Functional requirements

Table 9: UC category 3 - non-Functional requirements

ID	Description	Req. Level	Reporter	KPI
NF_UC3_01	APPLICATION DEVELOPER: CHARITY must use docker images to deploy applications (game servers).	MUST	ORBK	KPI-UC-3.5
NF_UC3_02	APPLICATION DEVELOPER: Mesh collider generator service should generate mesh collider (from 3d point cloud data) in less than 30s.	SHOULD	ORBK	KPI-UC-3.5
NF_UC3_03	APPLICATION DEVELOPER: Deployment API should response with game server IP after requesting for a new game server in less than 10 seconds after sending request.	SHOULD	ORBK	KPI-UC-3.5
NF_UC3_04	ADMINISTRATOR: Docker image storage must store up to 5 latest uploaded docker images.	MUST	ORBK	KPI-UC-3.5
NF_UC3_05	APPLICATION DEVELOPER: Docker image storage	MUST	ORBK	KPI-UC-3.5

ID	Description	Req. Level	Reporter	KPI
	must store docker images which size is up to 1GB.			
NF_UC3_06	APPLICATION DEVELOPER: Mesh collider generator service must generate mesh collider data which must contain vertices and triangles data.	MUST	ORBK	KPI-UC-3.5
NF_UC3_07	APPLICATION DEVELOPER: Mesh collider generator must take 3d point cloud data (list of Vector3) as an argument.	MUST	ORBK	KPI-UC-3.5
NF_UC3_08	APPLICATION DEVELOPER: Latency between client (player) and deployed docker image should be <100ms.	SHOULD	ORBK	KPI-UC-3.1
NF_UC3_09	APPLICATION DEVELOPER: Number of concurrent clients connected to game server should be ≥ 5.	SHOULD	ORBK	KPI-UC-3.3
NF_UC3_10	The time difference between the simulation state presented to each user should be <200 ms.	SHOULD	UTRC	KPI-UC-3.5
NF_UC3_11	The simulated environment must provide a consistent simulation state across all users, including rendering of other user activities.	MUST	UTRC	KPI-UC-3.3 KPI-UC-3.5
NF_UC3_12	The CHARITY orchestrator should automatically restart unresponsive components within 1 s of the component being declared unresponsive.	MUST	UTRC	KPI-UC-3.5
NF_UC3_13	The simulation should adapt imagery frame rate and resolution in accordance with available bandwidth, observed latency, and user equipment capabilities.	SHOULD	UTRC	KPI-UC-3.5
NF_UC3_14	The RTT from user action to presentation of updated imagery should be < 15ms.	SHOULD	UTRC	KPI-UC-3.2
NF_UC3_15	Number of concurrent users (virtual & real) in a single simulation scenario should be > 30.	SHOULD	UTRC	KPI-UC-3.3
NF_UC3_16	The video resolution of presented imagery must be greater than 60 FPS 4K.	MUST	UTRC	KPI-UC-3.5
NF_UC3_17	The CHARITY infrastructure must be able host containers based on the Docker image file specification.	MUST	UTRC	KPI-UC-3.5
NF_UC3_18	The CHARITY infrastructure must be able to provision NVidia GPUs for edge and cloud rendering (Kepler or newer), and support OpenGL.	MUST	UTRC	KPI-UC-3.5
NF_UC3_19	The CHARITY infrastructure should be able to provide access to arbitrary Internet-hosted services.	SHOULD	UTRC	KPI-UC-3.5
NF_UC3_20	The CHARITY infrastructure must be able to provide synchronized time services across all components, to <10 µs absolute error relative to UTC.	MUST	UTRC	KPI-UC-3.5

4 Conclusions

This Deliverable D1.2 summarizes the details of the UCs that will be used to test the CHARITY platform. These details were presented by the UC owners during the WP1 meetings, in the scope of T1.1 and include the reference scenarios, functional and non-functional requirements as well as the KPIs of the respective use cases. A comparison between the current and the envisioned state of the use cases highlights the benefits and impact of the CHARITY platform to the research areas related to Holography, Virtual Reality and Mixed Reality. The purpose of this deliverable is to provide a better insight of the basic components of every UC and their interaction within CHARITY platform; crucial requirements, that the project should provide, derive from comprehending these key elements. The list of use case functional and non-functional requirements will help shape and finetune components of the CHARITY architecture to help accommodate such applications. KPIs dictate the parts where extra caution and optimization is needed when designing both the platform and the UC respective applications. A deep understanding of the UCs will help identify potential issues and effort demanding tasks that should be addressed early enough, and therefore result in a risk-less continuation and completion of the project.