Bachelor’s thesis

Information and Communications Technology

2021

Mikko Österman

DEVELOPMENT OF A VIRTUAL REALITY CONFERENCE APPLICATION

TURKU AMK
TURKU UNIVERSITY OF
APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communications Technology

May 2021 | 42 pages, 4 pages in appendix

Mikko Österman

DEVELOPMENT OF A VIRTUAL REALITY CONFERENCE APPLICATION

The global pandemic has created a need for new ways of communication that are more immersive than the ones we have had before. Virtual reality sets the user in the middle of a virtual environment unlike any other technology before and could be used to alleviate the effects of current social distancing.

The purpose of this thesis was to study different virtual reality conference platforms and find out how much time and resources it would take to develop one for the School of ICT of Turku University of Applied Sciences (TUAS). The thesis also aims to explore some possible development solutions for different requested features of the application.

The work was carried out by first finding information and expert reviews about the most widely known virtual reality conference platforms and later comparing the functions of these platforms to the list of requirements that was given by TUAS. Expert interviews were also conducted to gain a better understanding on what is possible within the existing platforms and the requirements of creating a new virtual reality platform with a game engine.

Different feature implementations were also tested inside Unreal Engine 4 and Unity to determine how much time and effort would be needed to create a simple prototype with a single developer. These implementations were cut short halfway due to lack of time, and some parts were just left to speculation.

The thesis provides an overview on different options for virtual meetings and suggestions on what would be most suitable for the needs of TUAS School of ICT's needs. There was no single platform that had all the features the department wished for but the one that met most of the set requirements proved out to be AltSpace VR which the School of ICT had been using in the past.

The expert reviews and feature implementations gave an indication on what it would take to develop an entirely new platform suited for the TUAS School of ICT needs. It was generally agreed that such an undertaking would take a few years if it were to be fully polished and would require a team of multiple talented individuals. However, a simple prototype could be completed in a few months, but it would not include most of the requested features.

KEYWORDS:

Virtual Reality, Multiplayer, Unreal Engine 4, Unity

OPINNÄYTETYÖ AMK | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikka

2021 | 42 sivua, 4 liitesivua

Mikko Österman

VIRTUAALITODELLISUUDESSA KÄYTETTÄVÄN KOKOUSTILAN KEHITYSTYÖ

Globaalin pandemian aikana ovat etäyhteydet yleistyneet. Etäyhteyden kautta ei kuitenkaan saa samanlaista vuorovaikutuskokemusta kuin kasvokkain. Tämä on luonut tarpeen etäyhteyden immersiivisempään kommunikaatioon, jonka virtuaalitodellisuus mahdollistaa. Se asettaa käyttäjän keskelle virtuaalista ympäristöä tavalla, johon aiempi teknologia ei ole kyennyt, ja tätä voitaisiin hyödyntää lievittämään sosiaalisen etäisyyden aiheuttamia ongelmia.

Tämän opinnäytetyön tarkoituksena oli tutkia erilaisia virtuaalitodellisuudessa käytettäviä kokoustiloja ja selvittää, mitä sellaisen kehittäminen Turun ammattikorkeakoulun ICT-linjan käyttöön vaatisi. Opinnäytetyö tutkittiin erilaisia kehitysratkaisuja sovellukselta toivotuille ominaisuuksille.

Työ toteutetttiin hankkimalla aluksi tietoa ja asiantuntija-arvioita tunnetuimmista virtuaalitodellisuudessa käytettävistä kokoustiloista ja vertaamalla niiden ominaisuuksia Turun AMK:n antamaan vaatimuslistaan. Asiantuntijoita haastateltiin, jotta saatiin parempi käsitys siitä, mitä on mahdollista saavuttaa olemassa olevilla ohjelmistoilla ja mitä vaatisi kokonaan oman ohjelman kehittäminen pelimoottorin avulla.

Sovellukselta toivottujen ominaisuuksien toteutusta testattiin Unreal Engine 4:llä, jotta saataisiin parempi käsitys siitä, paljonko aikaa ja vaivaa yksinkertaisen prototyypin kehittäminen vaatisi yhdeltä kehittäjältä. Nämä toteutukset lopetettiin kesken ja osa toteutuksista jäi spekulaatioiksi.

Yksikään valmiista alustoista ei tarjonnut kaikkia Turun AMK:n ICT-linjan toivottuja ominaisuuksia, mutta kaikkein eniten vaatimuksista täytti AltspaceVR, jota he ovat käyttäneet jo aiemmin.

Asiantuntija-arviot ja ominaisuuksien toteutusten testaus auttoivat arvioimaan, mitä kokonaan uuden kokoustilan tuottaminen voisi vaatia. Yleinen näkemys oli, että ohjelman tekemisessä voisi hyvin mennä useampi vuosi, jos se viimeisteltäisiin huolellisesti, se vaatisi taakseen lahjakkaan kehitysryhmän. Toisaalta yksinkertaisen prototyypin tekeminen voisi järjestyä muutamassa kuukaudessa, mutta se ei pitäisi sisällään suurinta osaa toivotuista ominaisuuksista.

ASIASANAT:

Virtuaalitodellisuus, moninpeli, Unreal Engine 4, Unity

CONTENTS

APPENDICES

PICTURES

LIST OF ABBREVIATIONS

3D	Three-dimensional.
3D model	A three-dimensional representation of an object in a simulated environment.
AI	Artificial intelligence.
AR	Augmented reality. A situation where Virtual objects are augmented into the real world.
Avatar	A digital character possessed by the user or AI.
Bot	An AI controlled avatar or an entity in a text messaging platform.
C++	A programming language.
Framework	A collection of programming libraries that make it easier to develop a certain type of software.
ICT	Information and Communications Technology
P2P	Peer-to-Peer. A networking architecture model.
PC	Personal Computer.
RPC	Remote Procedure Call
SDK	Software Development Kit
TUAS	Turku University of Applied Sciences
UE4	Unreal Engine 4. A game engine developed by Epic Games.
UI	User Interface
Unity	A game engine developed by Unity Technologies.
VoIP	Voice over Internet Protocol. A way of communicating with voice over the internet. A user’s speech is recorded, packed and sent to the other user’s audio listener over the internet connection.
VR	Virtual Reality. A simulated environment that is accessed with a virtual reality headset.
XR	Extended reality. The X in the abbreviation can be replaced with A as in augmented or V as in virtual depending on which case of extended reality is being utilized. Varjo uses XR in their product name to demonstrate that the device uses both.

1 INTRODUCTION

The global pandemic has created a necessity for different online meeting platforms as people feel the need for more immersive experiences. One way of offering these immersive experiences is Virtual Reality or VR which brings the user in to a virtual world by utilizing VR headsets that cover the user’s eyes. There are already many VR applications that are accessible by multiple users simultaneously. These will be referred to as Virtual Reality Conference applications in this thesis due to a lack of more standardized expression. Virtual Reality Conference applications are being developed by companies such as Microsoft, Facebook, Fake, Mozilla and many others.

The thesis topic was chosen because industry giants such as Microsoft and Facebook are making huge investments in the field and there is not much scientific material available that studies these applications. Reliable scientific research was not found on some of the platforms at all and it should be noted that this study had to rely on non-scientific articles when studying them.

The purpose of this thesis is to study if it is more feasible for a conference organizer to use these existing platforms or create a new one by using a game engine or some other development framework.

The thesis consists of two parts, the theoretical part which gives an in-depth view on some of the more well-known existing platforms. The features and challenges they present to the conference organizer are also explored and compared to the challenges the game engines have regarding the creation of such platform. To give a better perspective on these issues, a few experts were interviewed who offered their unique views on the development challenges and what is achievable with the existing platforms.

The practical part of the thesis uses the information acquired from the first section to study if it would be possible to create a new conference platform that would be more optimal for the organizer than the existing platforms. The engine that was chosen for these tests is Unreal Engine 4 to give more options for development environments as most of the staff in TUAS school of ICT are already familiar with Unity but very few have tried UE4.

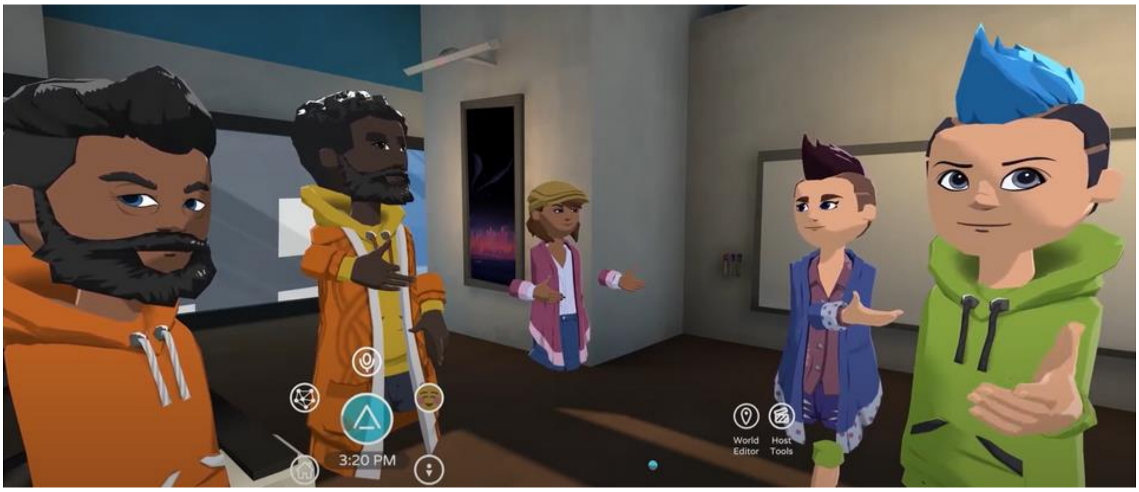
2 EXISTING VIRTUAL REALITY CONFERENCE PLATFORMS

Several virtual reality conference platforms already exist and some of the most well-known cases are studied in this section. The Helsinki XR Center has already carried out considerable research on different platforms and their data will be utilized in analyzing the platforms. Their study focuses mainly on platforms usage as a remote working space so information about the usage on larger events was added on top of their results. (Helsinki XR Center, 2021)

Scientific articles and studies on the subject were hard to find and research had to rely on the documentation of the virtual reality conference platforms and non-scientific articles.

2.1 AltspaceVR

AltspaceVR was launched in 2015 and was later acquired by Microsoft in 2017. It is currently one of the most prominent social VR platforms and currently the only one that can handle over one hundred users watching the same presentation which makes it ideal for large events. It is being developed by Microsoft which means there should be significant resources behind the development as they believe VR will be the next big influencer platform. The application is available for free through Steam or from Windows Store. (Helsinki XR Center 2020a; Matthews et al. 2020)



Picture 1. AltSpaceVR visual style and UI (Microsoft 2021).

Creating an event in Altspace requires registration at their website after which it is possible to schedule an event. An event can be set as private so that only people with a password can enter or as public which is free for all. (Microsoft 2021a)

The tools available to a user focus around presentations and talk programs. Events can have host and moderator roles which provide some tools that regular users do not have. Host generally means the person who created the event and was the one that started the application in the chosen environment. (Microsoft 2021b)

The enterprise edition of AltspaceVR is built on Microsoft Mesh, which means that there will soon be more 3D photorealistic presence and spatial multiplayer experiences. Mesh is a mixed reality application that is designed for collaborative work with augmented reality. This combination gives AltspaceVR great potential for collaboration and corporate events in the future. (Roy 2021a)

2.2 Horizon

Facebook Horizon takes a more creative approach to virtual reality meetings by including games and environment modification options to users. Horizon launched early 2020 in a closed beta testing which is still going on. According to a review by TechCrunch, the application seems to be a combination of the virtual reality pioneer Second Life, AltspaceVR and PlayStations gamified creation tool Dreams. (Constine 2019)

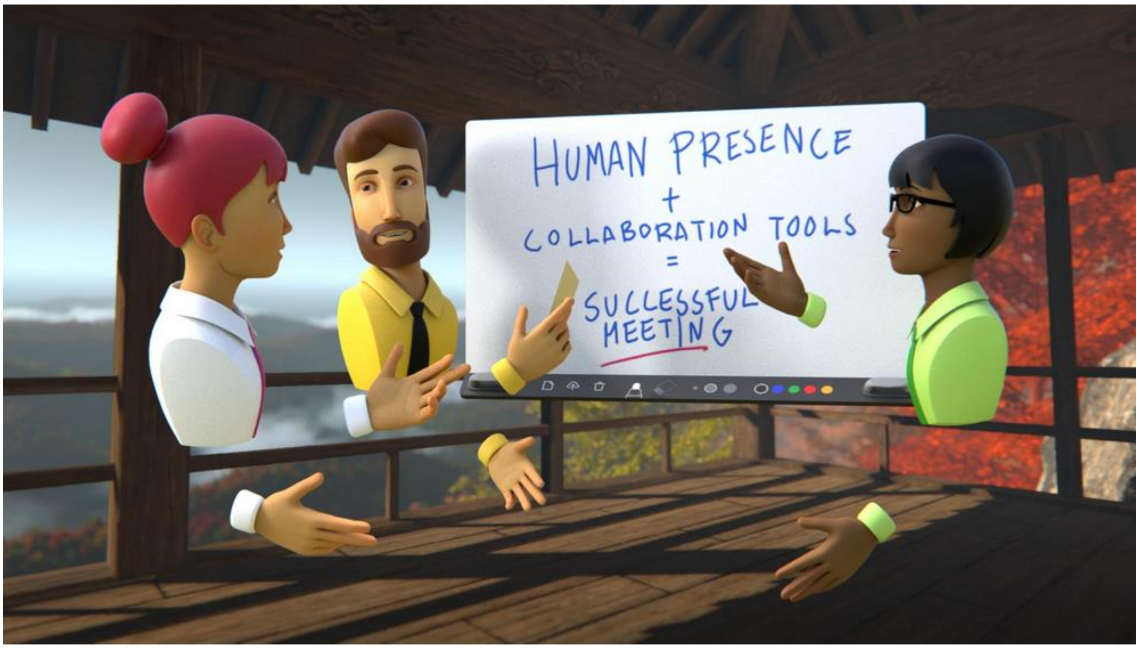
Picture 2. Horizon visual style (TechCrunch 2020).

Horizon focuses more on organizing the users to smaller rooms instead of one large space. Everything within the environment is built with integrated creation tools that the users can utilize to create their own environments, 3D models and even some basic scripting. (Lang 2020)

In summary, the Facebook Horizon is targeted for everyone as a recreational space which tries to bring the novel Ready Player One into reality. It is intended to be used with a limited number of users in same space and at its current state would not be ideal for large conferences.

2.3 Glue

Glue is a cross platform virtual space that offers possibilities for screen sharing, note taking, presentations and viewing of videos and 3D models. The different environments are made by the developers or created by the users and approved by the Glue developers. Users can speak to each other with the built in VoIP-system. A variety of different VR devices is supported, and the application can also be accessed from desktop. The tools the user has available are dependent on the device they use to access the app. According to the Helsinki XR Center’s tests, screen sharing is only available for desktop users. Glue requires the installation of the application which is available for iOS, Windows and Oculus Quest. Max 30 simultaneous users. (Helsinki XR Center 2020b)



Picture 3. Glue visual style (Glue 2021).

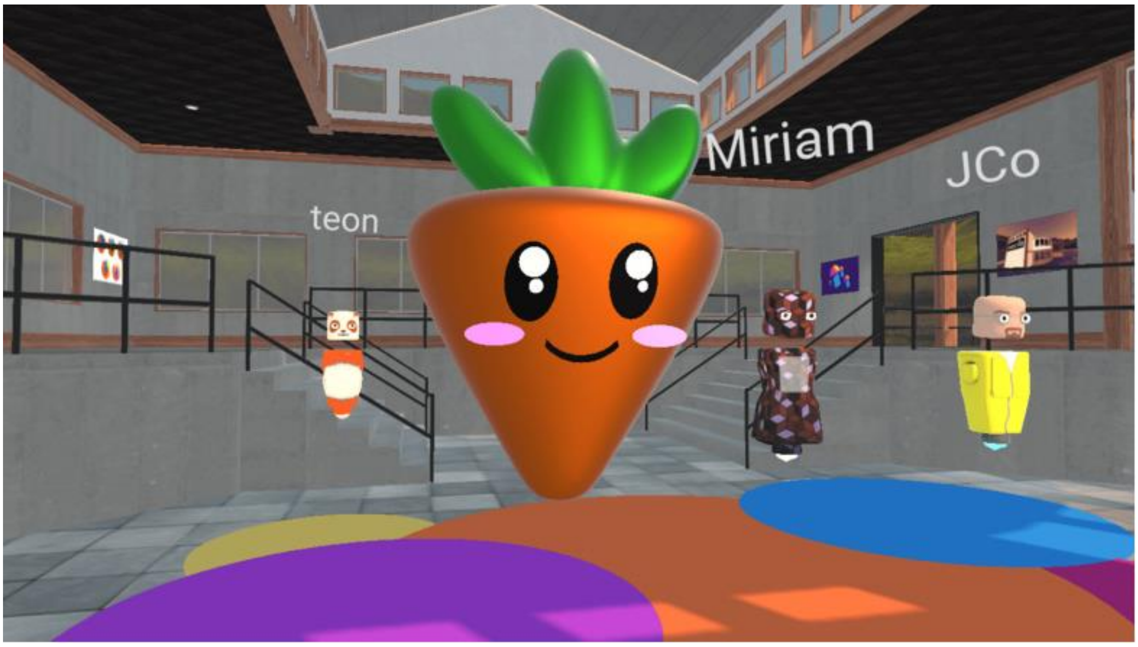
Testers felt more presence with their coworkers in this environment than with Teams or Skype. The visuals and the audio quality were praised, and the different tools were useful and fun according to the test group. (Helsinki XR Center 2020b)

User tests were also conducted by Niina Järvinen for her Business Information Technology thesis in 2019. These testers found the application interesting and easy to use although no instructions or guidance were given to the testers. Surprisingly the strongest positive reactions about the application came from the testers that were already familiar with similar applications. (Järvinen 2019)

Meeting participants are represented via 3D avatars that track the participant’s hand movements, eye movements, and voice. This is achieved by the fading of attenuation audio volume as the distance between user avatars increases. As a result, users experience very natural audio as well hand and some facial gestures which all add up to the immersion. (Roy 2021b)

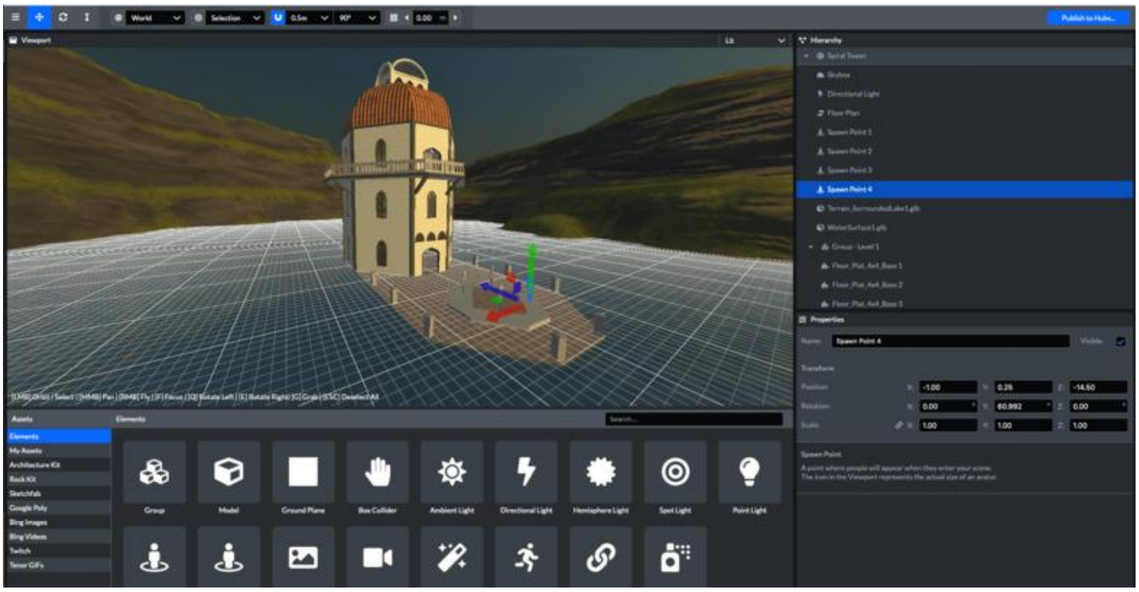
2.4 MozillaHubs

“Mozilla launched Mozilla Hubs in 2018, as a free, experimental software that’d help users from different industries and verticals try out VR collaboration as an alternative to traditional conferencing tools.” (Roy 2021c)



Picture 4. MozillaHubs visual style (Mozilla 2021).

MozillaHubs is an open-source virtual space that does not require any software installation as it is run from a web browser. It is also available for free and supports all devices: VR headsets, desktop, tablet and mobile. These properties make MozillaHubs the most accessible virtual environment of the ones covered in this thesis. Being open source means that the application is very customizable, and the users can upload their own avatars and build their own environments. Mozilla has even made a separate 3D editor for this purpose called Spoke. The created rooms are private and only people who have access to the URL can join the room. (Helsinki XR Center 2020c)

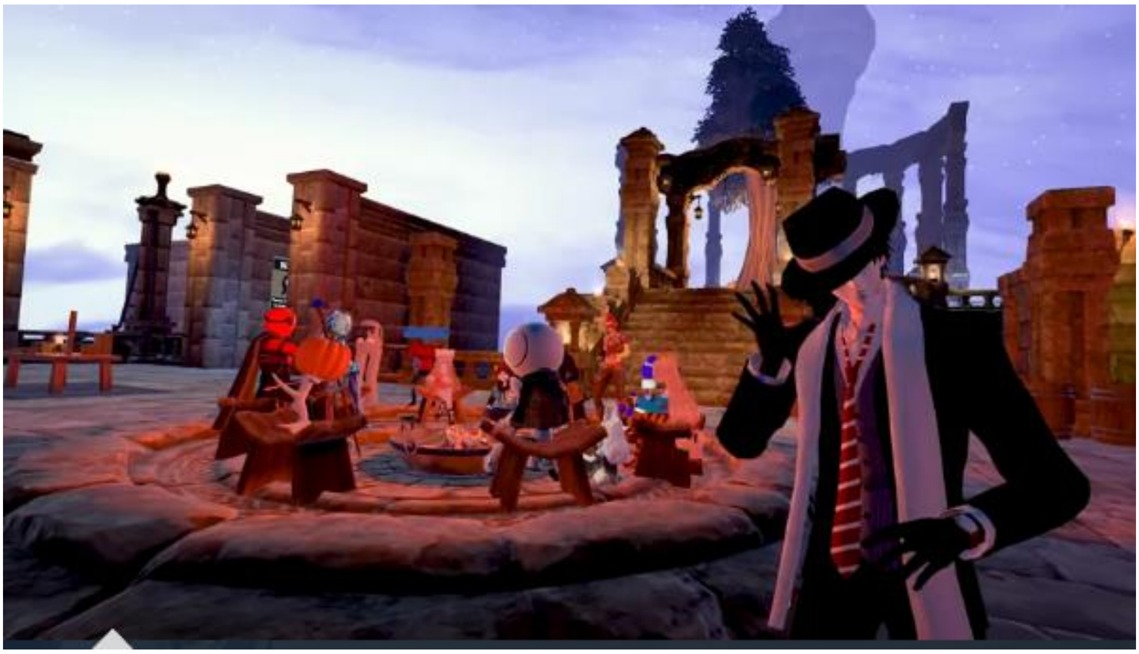


Picture 5. Spoke UI (Mozilla 2021).

The Helsinki XR Centers testers found the MozillaHubs to be very fast to use as no installation or downloads were required to get started. Surprisingly, the testers found that the app was easier to use on mobile devices than with VR headsets. Mozilla warns users that although several devices are supported, the experience might differ drastically and they do not recommend more than 10 people in the same space with mobile devices. It is worth to note that at its current state the platform does not allow more than 24 users in the same space regardless of what device is being used. (Helsinki XR Center 2020c)

2.5 VRChat

VRChat is the most customizable virtual reality platform of the ones covered in thesis. Its community has created hundreds of avatars, worlds, and activities for everyone to enjoy. VRChat supports PC VR headsets, Oculus Quest and is also usable with desktop computer but available features vary between devices. The Helsinki XR Center testers found out that some worlds that claimed to have Quest support did not actually support all the same features as other devices. The testers did not think the platform was at its best as a work environment, but it was great for recreational purposes and social gatherings. They found their experiences in VRChat refreshing, fun and something they will not forget any time soon. (Helsinki XR Center 2020d)



Picture 6. VR Chat visual style (Steam 2021).

According to an article from Mashable, the VRChat is an absurd experience with its own community and that is the beauty of it. It allows the users to live their childhood dreams of doing silly things without consequences and they describe the app as “just dumb fun”. (Volpe 2020)

3 DEVELOPMENT REQUIREMENTS AND CHALLENGES

Turku University of Applied Sciences School of ICT has requested specific features that the virtual conference tool should have and some that they would like to have but are not mandatory. Finding the best platform for the conference application is done by comparing the platform to these requested features. The goal is to find the platform that checks out most of them and then compare would it be feasible to create an entirely new application or is it possible to customize the existing one to the department's needs.

The School of ICT wants the conference center to have customizable avatars. This means that the user should be able to customize the appearance of the character they control. They also want there to be an auditorium for keynote speakers with capacity for as many listeners as possible. There should also be an exhibition hall with different lecture and meeting rooms, display stands, posters, video demos and 360 photo spheres. Private meeting or a voice channel was also requested with the option to have video calls inside the app. The atmosphere and environment should also be customizable to create a fitting mood for different events and that environment should also include a map so that users can find the different rooms.

The optional features that the department feel would be nice to include are user access control, an automated chatbot and feedback system. Utilizing the forementioned would require artificial intelligence.

Based on our understanding one thing that no product in the market has yet is the possibility to have product presentations done with extended reality although Microsoft has announced that it will be bringing that feature to AltspaceVR in the future (AltSpaceVR 2021.)

The information acquired for previous chapter was utilized in finding out the requirements and challenges for development. Experts on AltSpaceVR and Unity development were also interviewed to get an opinion from professional software developers. Unfortunately, there was no expert available that would have had similar amounts of experience in UE4 development and information was acquired from personal experience and other external sources.

There are many other game engines and frameworks that could be utilized to create a virtual conference center. Unfortunately, experts on different frameworks could not be reached within the schedule of the thesis for an interview and this aspect has been left unexplored.

3.1 AltspaceVr

To get a better understanding on what it really is like to create custom environments in AltSpaceVR expert interviews were conducted. Jami Aho is a graphical artist currently employed by TUAS who has a strong background in 2D and 3D graphics. Panos Markopoulos is a game designer who is also employed by TUAS and has a strong background in game design and virtual reality. They were both included in the team that created a custom meeting room in AltSpaceVR for an event.

They chose AltSpaceVR for their platform since its process was straightforward especially since they were both familiar with Unity. The team created a scene in Unity and imported it to AltSpaceVR with all the assets included. The biggest problems were faced when they imported videos since those needed to be edited individually to a smaller format and added separately in the scene. They tried using just links to external video players but that caused serious performance issues and turned the whole application unusable.

One thing the team found missing was the ability to add custom scripts in their scene. All custom scripts need to be packaged into an SDK and sent to AltSpaceVR for review before those could have been used for development. There exists a selection of preapproved SDKs that could be utilized when creating such a scene, but the collection does not cover everything. Aho and Markopoulos both explained that this was because some scripts could be used for malicious intent and AltSpaceVR wants to make sure no harm is done to the users.

According to Aho and Markopoulos AltSpaceVR can support large crowds of people by splitting the crowd in different rooms and mirroring one speaker or a video to all the rooms. However, they noticed that videos are not in sync between the users and the playback starts from the beginning for each client once they enter the room. The host has the option to move users between the rooms if needed.

Out of the features that TUAS ICT-department requested the photos and 360-degree photospheres were very easy to implement according to the team. Photospheres were implemented by just inverting a 3D sphere objects material and placing player inside of it. This solution worked for photos and videos.

Private meetings and video calls are possible inside AltSpaceVR but they both require extra effort from the hosting administrator. The host needs to setup the rooms accessibility for specific usernames manually which can be done in runtime, but the changes take about one minute to take effect. Therefore, setting up a private meeting with a new contact is not very convenient. Video calls are handled by the host as well and he or she needs to grant administrative rights to each person that would like to start broadcasting their video.

Aho and Markopoulos both agreed that out of the available platforms AltSpaceVR was the one that fitted their needs the best although it had its limitations.

3.2 Glue

Glue has its user capacity limited to 30 simultaneous users which does limit its usability for large events. They provide any tools to customize the environment to same extent as AltSpaceVR and even more. Panos Markopoloulos found Glue to be superior when compared to AltSpaceVR in usability, features and even in visual presentation. There is also the possibility to use your own custom-made scripts once you run them through Glue’s inspection first. The big downside according to Markopoulos is that every user needs to have a paid subscription to their service which skyrockets the costs very quickly in bigger events.

3.3 Unity

The two most popular game engines in the market are Unity and Unreal. They both have a lot of in-built functionality that makes game development easier and faster in most cases.

Sami Laukkanen was interviewed to get a better understanding on Unity virtual reality multiplayer development. He is a project engineer in TUAS who has been doing programming and development with Unity mostly focusing on virtual reality and lately with a multiplayer aspect included. He recently created a virtual reality multiplayer teaching environment where the teacher hosts a game and students join as clients. Teacher shows how a fire should be extinguished and students try to do the same.

Laukkanen used Photon Bolt for the project he made and mentioned during the interview that the maximum number of users could be as high as 64. They did not do any stress testing however and the highest number of simultaneous users was three. According to Laukkanen the paid subscription of Photon Bolt could handle 5000 simultaneous users in a server but not in a same room. According to the pricing chart in Photon website the plugin supports 16 users in a same room and a maximum of 50.000 simultaneous users on a server.

Unity currently lacks built-in multiplayer functionality and relies on plugins instead. The most common plugin for multiplayer is Photon Bolt and Photon Pun which provide access to both client-server and peer-to-peer architecture (Photon 2021).

Client-server architecture means that most of the actual gameplay and processing is done on a single computer called server and the other users rely on it for information. This kind of architecture tends to be quite safe since the server is making sure that the information is correct. The downside of client-server architecture is that it is very heavy on the server computer if there are a lot of connected users called clients. This problem is more considerable when users are wearing VR headsets as the motion controllers and the headsets tend to always pass a lot of information. To handle a large crowd of VR users their past information needs to be cut down by a significant amount or some other type of networking architecture needs to be utilized.

Peer-to-Peer or P2P architecture sends information between all the users when needed and would be an excellent way of reducing the stress on a server. Unfortunately, Photon’s plugins support P2P architecture only on game sessions where there are two players.

There is another plugin for Unity called Mirror that excels in games with large amounts of users and has a showcase with 480 simultaneous users in a same space. Mirror can also handle VR, but tutorials are not as easily available as they are for Photon’s plugins. This lack of tutorials could hinder the development as a lot of time needs to be sacrificed for trial and error. Laukkanen did not test Mirror in his project mainly because Photon Bolt was better documented, and more guides were available.

The key thing to consider when developing a virtual reality conference application would be creating the tools to allow users with little technical background to create their own environments according to Laukkanen. He thinks that while the mechanics in such an application would be simple since not many things are replicated between users, a major undertaking would be the backend with servers and access handling.

Laukkanen stated that if he were to develop one all by himself it would take him a couple of years due to his lack of skill in the backend side of such applications but with a bigger team with required specialists a working prototype could be done in four months. By utilizing all available resources in their work community, he estimates that a fully polished version could be achieved in a year. Markopoulos and Aho had a less optimistic opinion as they believe it would take several years to create a polished version even with a big team and lots of resources. They all agreed that since a similar project has not been made in their workplace, unforeseen circumstances might delay the process significantly.

3.4 Unreal

Unreal has built-in multiplayer functionality that uses client-server architecture. The engine has an open-source code base that can be modified to create P2P multiplayer functionality for some parts in theory at least. According to forum discussions these kind of modifications to the engine would be very difficult and P2P is not recommended to be used in multiplayer and developers are encouraged to use dedicated servers instead. Using dedicated servers with Unreal limits the player cap to 100 simultaneous users which could be multiplied with the same kind of workaround that AltspaceVR is doing by mirroring a single performance to different rooms. (Unreal Engine Forums 2019)

UE4 is widely regarded as a reliable and scalable solution when it comes to big projects. It is also a massive collection of different tools and navigating inside the engine and its features takes some time to get used to. The engine forces the developers to use a certain architecture when making applications which allows the scalability especially for the developers who are unfamiliar with software architecture. These reasons make the UE4 a strong candidate to be chosen as the development platform if the intention is to create a completely new virtual reality conference platform.

Downside of UE4 development is the fact that not many developers currently working for TUAS are familiar with the engine and it would take some time to first learn how to use it. The documentation is also less extensive when compared to Unity’s which also has wider and more beginner friendly community.

4 DEVELOPMENT WITH UNREAL ENGINE 4 AND UNITY

After a study of different possible frameworks for creating the Virtual Conference Center, it was decided to look more in depth what the requirements would be if it were made with Unreal Engine 4 and Unity. The main reason was that they are two most widely used game engines and both provide enough built-in features to ease some of the burden. There was also no existing information within TUAS on UE4 VR development with multiplayer so it was an opportunity to research if Unreal would be a suitable development environment for such applications. Since there was very little previous experience in UE4 development, it was decided to utilize the visual scripting system called blueprints instead of writing code in C++. With Unity the scripting was done with C# and a previous project made by Sami Laukkanen was used as a starting point to save time.

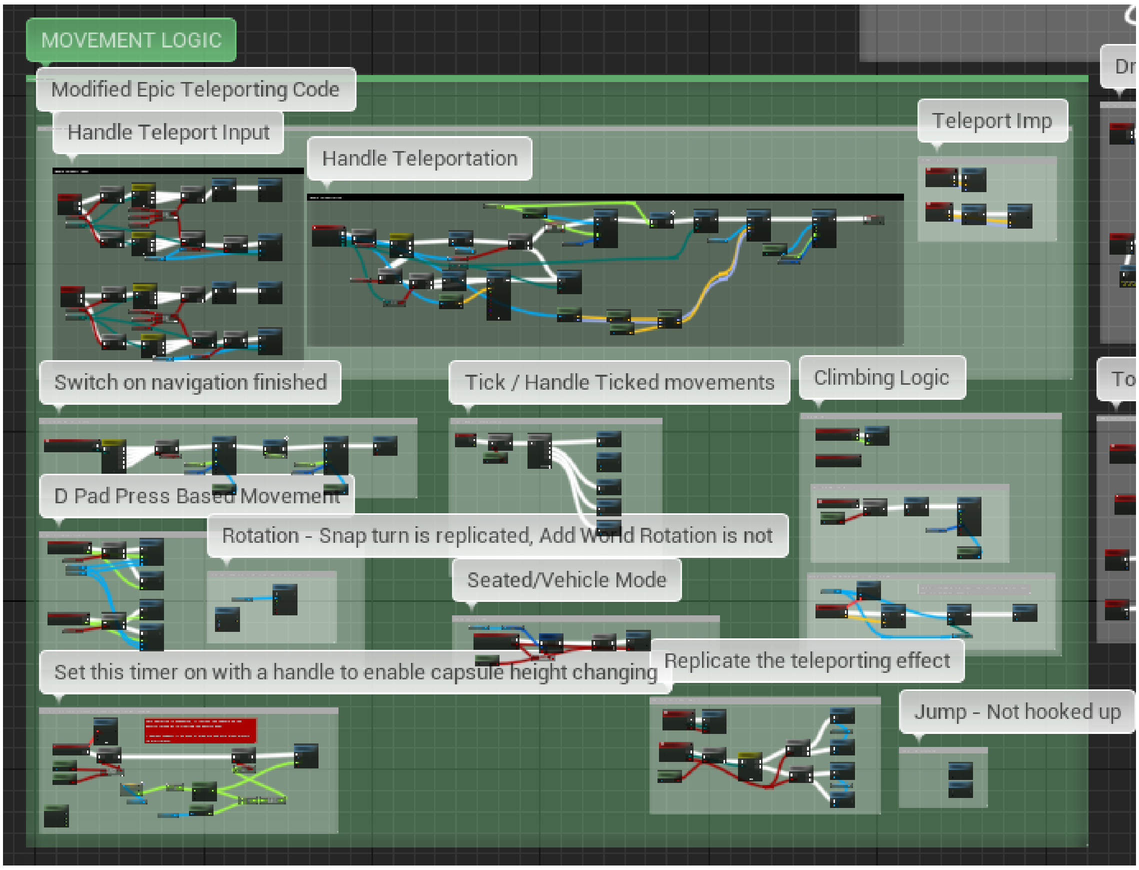
Due to the limited timeframe of the thesis some parts of the development were left untested and only theoretical solutions were applied to the given problems. The virtual reality mechanics, multiplayer connection, VoIP, slide show presentations, video player and text chat were tested while the rest of the functionalities were left at design stage. If TUAS school of ICT wishes these tests could be continued further later.

4.1 Virtual Reality mechanics

Unreal Engine 4

UE4 has a template that has a lot of built in VR mechanics that gave a great head start for the development. The template includes working functionality for tracking the motion controllers and headset location which means that the application displays the position and rotation of the players VR headset and VR controllers inside the virtual world depending on their position in the real world. A template for a grabbable object is included as well. A user can grab an object within a virtual world using a controller and the objects position is set to follow the controller’s movements. Player movement by teleportation or locomotion with joystick are also available inside different character templates. Teleportation is executed by pressing down the joystick which toggles an arc that can be aimed at the ground. Releasing the joystick initiates teleportation to the position that the arc was pointing at. Locomotion movement lets the user use the joystick directly for movement. If the user turns the joystick to the left, the character moves to the left and so on. Locomotion is rarely used for movement in VR applications mainly because it causes nausea for some users. There is also a laser pointer that can interact with 3D user interface objects. A laser pointer is attached to the user's hand in virtual world and if pointed at a user interface object and a trigger is pulled in the controller a similar event is triggered as if the element were clicked with a mouse cursor. All of these come with an example project that showcases all the features.

There is also a plugin called VR Expansion available which adds even more features to start with. The plugin has implemented features for climbing, more variations for player movement, interactable 3D object buttons, more refined grabbing system, vehicle driving and completely overhauled VR character that is built with networking in mind. VR Expansion is available for free and it has been utilized in creation of 29 published commercial products. (VR Expansion Plugin 2021)


MOVEMENT LOGIC
Modified Epic Teleporting Code
Handle Teleport Input
Handle Teleportation
Teleport Imp
Dr
Switch on navigation finished
Tick / Handle Ticked movements
Climbing Logic
To
D Pad Press Based Movement
Rotation - Snap turn is replicated, Add World Rotation is not
Seated/Vehicle Mode
Set this timer on with a handle to enable capsule height changing
Replicate the teleporting effect
Jump - Not hooked up

Picture 7. An overlook of the VR Expansion player movement logic blueprint.

Unity

Virtual Reality mechanics were already implemented in the project that was used as a base for the development. The original project uses SteamVR-plugin to handle most of the VR mechanics although Sami Laukkanen has made some customization on some of the mechanics. SteamVR covers basically the same features as Unreal VR Template does and Laukkanen’s modifications add some of the features that the UE4 VR Expansion has such as climbing and a more refined grabbing system. It should be noted that the modifications have also been made with networking in mind.

4.2 Multiplayer

Unreal Engine 4

To find out how difficult it is to implement multiplayer to a VR application is, it was first tested on the UE4’s own template without the plugin. Later it was tested with the plugin and results were compared.

Adding multiplayer functionality to UE4’s VR template took time since there were a lot of things that were built for single player only. Some parts of the blueprints used a hard coded player index 0 which is fine if there is only one player but finding and replacing all those references took some time.

Hosting a game session and joining it proved out to be simple task and implementing it was very quick. A simple UI was designed to help with the testing.

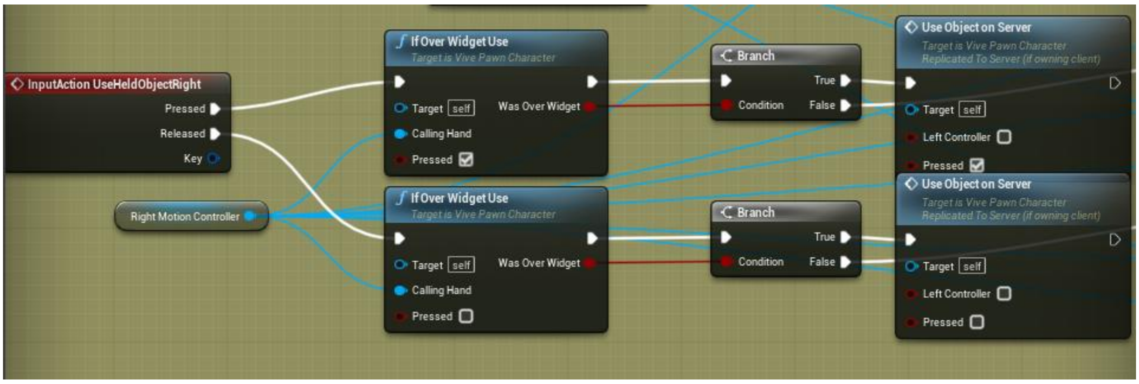
After some simple setting up of the functions the client side could see the movements of the host, but host saw only a standing still 3D model of the client. Later, updating of the client’s position after teleportation was achieved but the real time movement was not achieved during the given timeframe. It appeared that replicating values from host to client worked but not the other way around. The problem was most likely within the ownership of the objects within the scene as only the user who owns the object can modify its values and broadcast them to other users.

Trying to get UI buttons to function in multiplayer took a lot of trial and error until the right solution was found and it seemed that the problem was gone in seconds. The UI functioned well in the editor after the fix but when tested with two computers and headsets it was noticed that the UI updates only from host to client and not the other way around. A lot of time was spent trying to fix this, but no solution was found. The problem seemed to lie within the ownership of the actors within the level which could be solved but would take a lot of time and reworking some classes within the template.

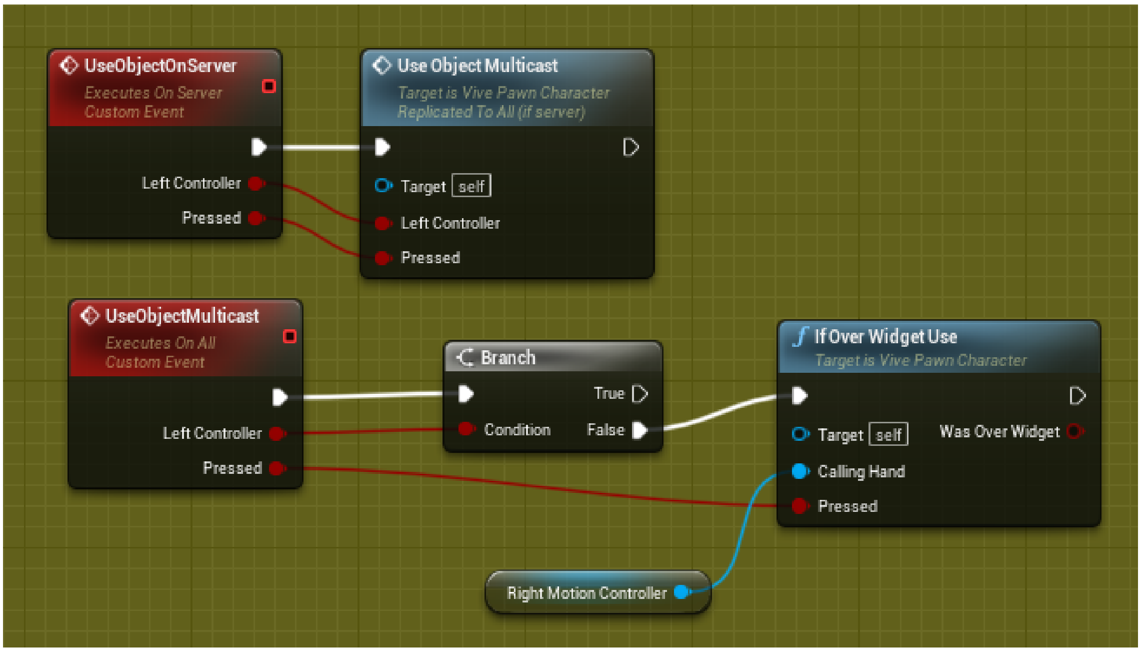
Implementing multiplayer on top of UE4's template proved out to be very difficult and unreliable solution. After extensive study in Unreal Forums, it was clear that the only way to get it working would be to recreate some of the template's functionalities. The result for this experiment was a version where the players positions could be replicated between users after teleportation and the host's interactions with UI elements were replicated to the clients.

VR Extension had all the functions already implemented except for UI interactions. The plugin has its own thread at the Unreal Forums and the author commented there that getting the UI to replicate would be too much work and will not be implemented any time soon. This information suggests that workaround solution would be preferrable in this case. (Unreal Engine Forums 2021)

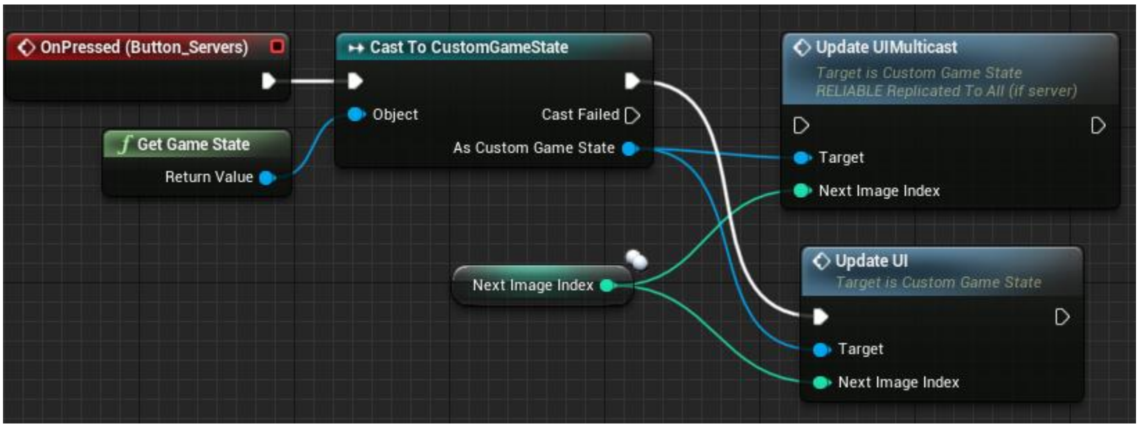
The UI interaction was implemented on top of the VR Extension plugin by adding functionality to an existing function. When UI Widget is clicked the function calls RPC that sends the current image index integer from user to server that broadcasts it back to all clients. The integer then determines which image is displayed in the UI Widget.



Picture 8. Input action triggers when button is pressed and released.



Picture 9. RPC sends data from client to server which broadcasts it to all clients.

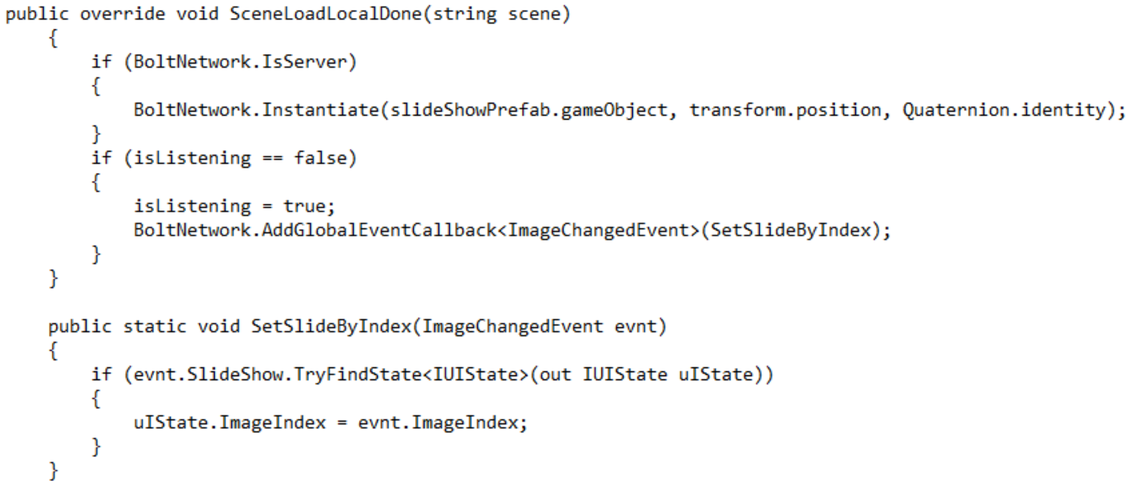


Picture 10. Pressing the Next Slide -button triggers the UpdateUI -function.

Unity

The base project had most of the multiplayer functionality already implemented but as in UE4 there were no functionality for UI interactions. In this case the original project used Photon Bolt plugin to handle networking as Unity currently has no built-in networking functionality.

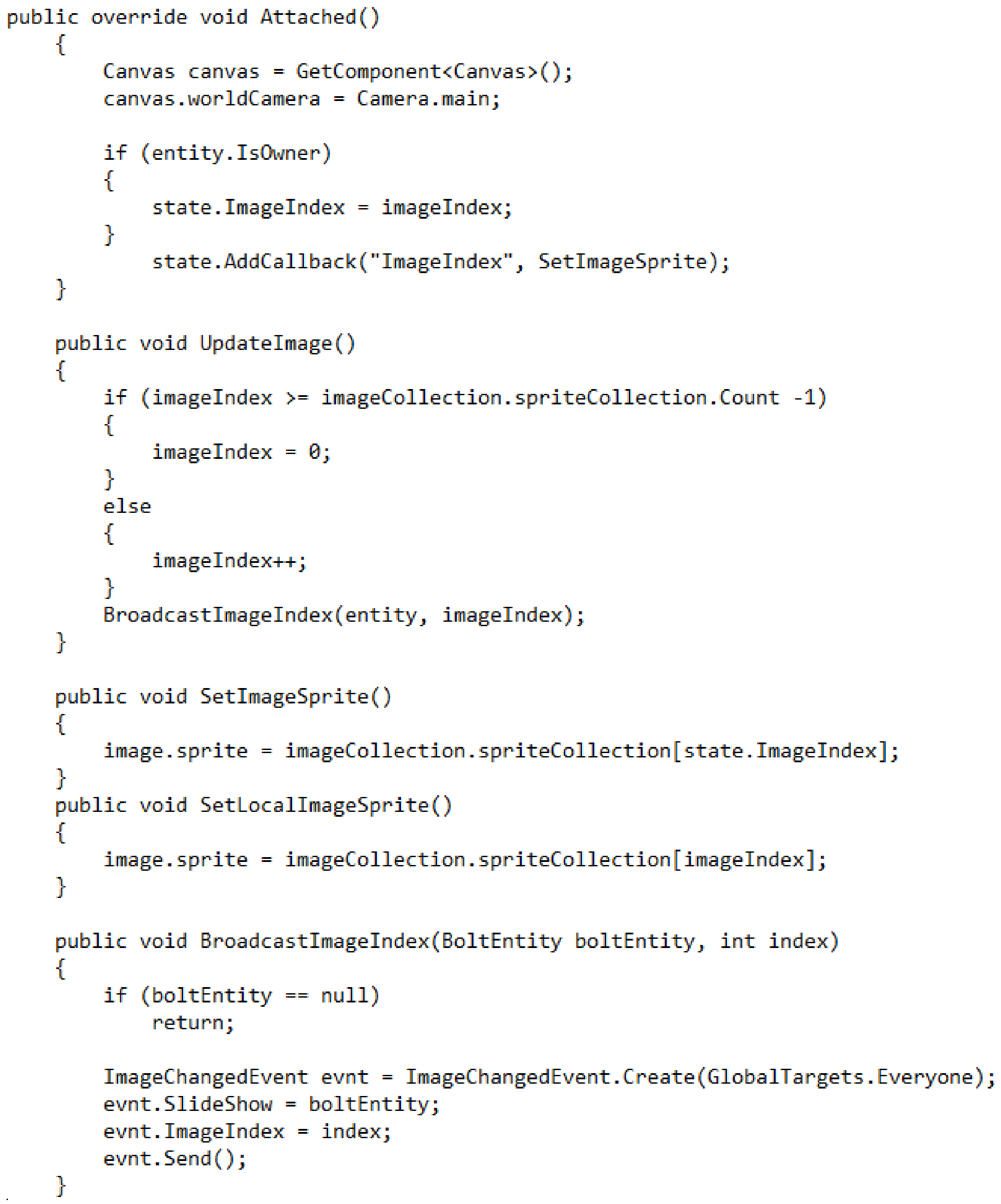
A slide show was implemented by creating a Photon Bolt asset called UIState that contains the current image index integer for the slide show and a ImageChanged event that broadcasts the integer to specified users. Once the integer in UIState is changed it triggers a function that sets the correct image to the UI by index integer.



```
public override void SceneLoadLocalDone(string scene)
    {
        if (BoltNetwork.IsServer)
        {
            BoltNetwork.Instantiate(slideShowPrefab.gameObject, transform.position, Quaternion.identity);
        }
        if (isListening == false)
        {
            isListening = true;
            BoltNetwork.AddGlobalEventCallback<ImageChangedEvent>(SetSlideByIndex);
        }
    }

    public static void SetSlideByIndex(ImageChangedEvent evnt)
    {
        if (evnt.SlideShow.TryFindState<IUIState>(out IUIState uIState))
        {
            uIState.ImageIndex = evnt.ImageIndex;
        }
    }
```

Picture 11. SlideShowSpawner -functions.



```
public override void Attached()
    {
        Canvas canvas = GetComponent<Canvas>();
        canvas.worldCamera = Camera.main;

        if (entity.IsOwner)
        {
            state.ImageIndex = imageIndex;
        }
            state.AddCallback("ImageIndex", SetImageSprite);
    }

    public void UpdateImage()
    {
        if (imageIndex >= imageCollection.spriteCollection.Count -1)
        {
            imageIndex = 0;
        }
        else
        {
            imageIndex++;
        }
        BroadcastImageIndex(entity, imageIndex);
    }

    public void SetImageSprite()
    {
        image.sprite = imageCollection.spriteCollection[state.ImageIndex];
    }
    public void SetLocalImageSprite()
    {
        image.sprite = imageCollection.spriteCollection[imageIndex];
    }

    public void BroadcastImageIndex(BoltEntity boltEntity, int index)
    {
        if (boltEntity == null)
            return;

        ImageChangedEvent evnt = ImageChangedEvent.Create(GlobalTargets.Everyone);
        evnt.SlideShow = boltEntity;
        evnt.ImageIndex = index;
        evnt.Send();
    }
```

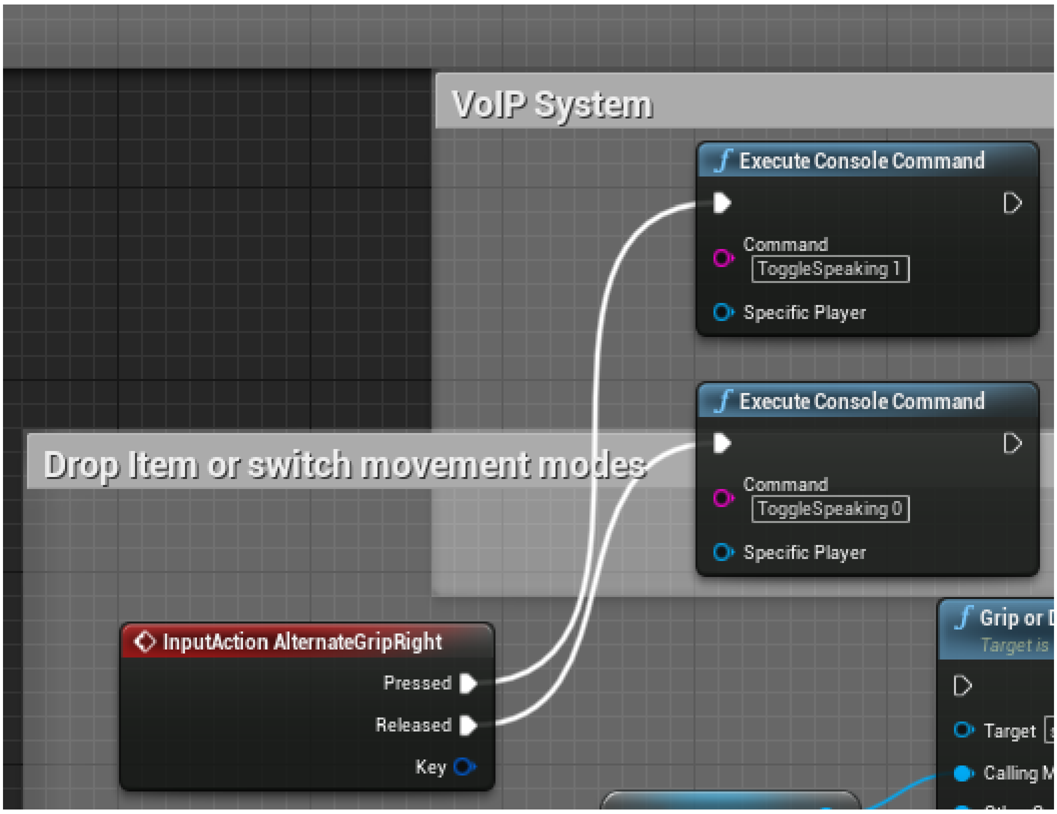
Picture 12. SlideShow -script functions.

]

4.3 Voice over Internet Protocol

Unreal Engine 4

VoIP is handled in a way that once a certain button is pressed the application opens the channel and user's speech will be sent to the other user while the button remains pressed. Releasing the button closes the channel and no speech gets sent. It was very quickly implemented into the project as it took only a few functions and some changes to the project settings. The VoIP functionality was achieved by following a guide from Couch Learn. (Matt 2019)



Picture 13. InputAction that triggers VoIP on when pressed and off when released.

Unity

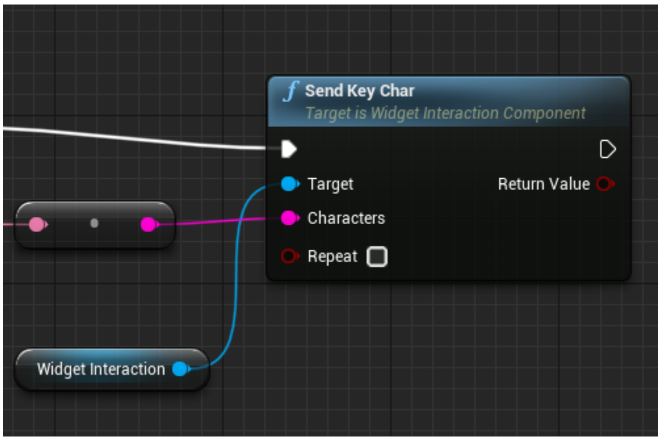
The base project had VoIP already implemented and the user does not need to press any buttons as the channel is always open. The implementation was done by using Photon Voice plugin which requires only dragging the component in to the scene and adjusting the settings if needed. Photon Voice is included in the Photon Cloud package and is free to use if there are less than 20 connected users.

4.4 Messaging and Video player

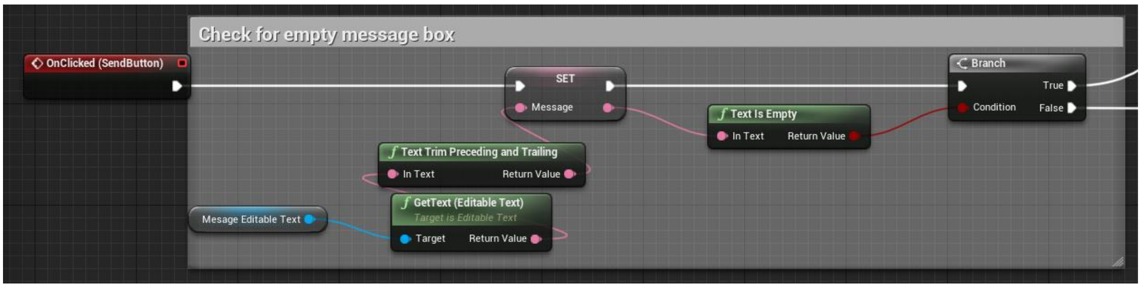
Unreal Engine 4

Text messaging between users would be quite straightforward and lots of guides are available that teach how it could be implemented. Usually, text messaging is implemented on the user’s screen as a scrollable text box that shows the past messages written by users. If the messaging system needs to be more complex with different channels it would be easier to use a plugin like Complete Chat System that is available in Unreal Marketplace. Text messaging would be hard to use while wearing a VR headset and it is something that needs more consideration whether to include it in the project at all. Adding a text box to a user’s screen in VR is not a good idea since the user would either have a hard time trying to see what is on there or it would be too much in the view constantly. (Unreal Marketplace 2021)

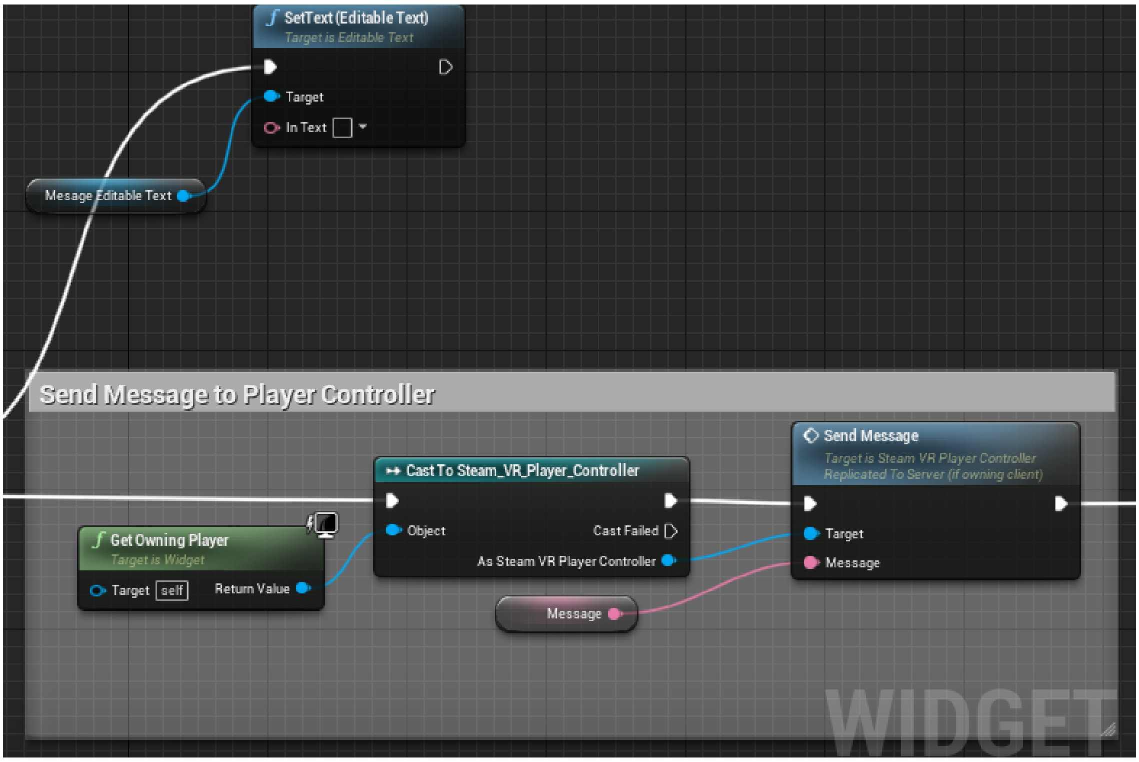
A chat system was implemented in the project by utilizing a simple chat system created by another developer and linking it to the player controller and a virtual keyboard from VR Extension. A functionality to send characters from a string was added to the keyboard blueprint to send the text to the chat. The outcome was a 3D canvas with a text chat and a virtual keyboard in the virtual environment and not attached to the screen. A text-to-speech and speech-to-text functionalities and suggested phrases could be the next step if this were to be developed further.



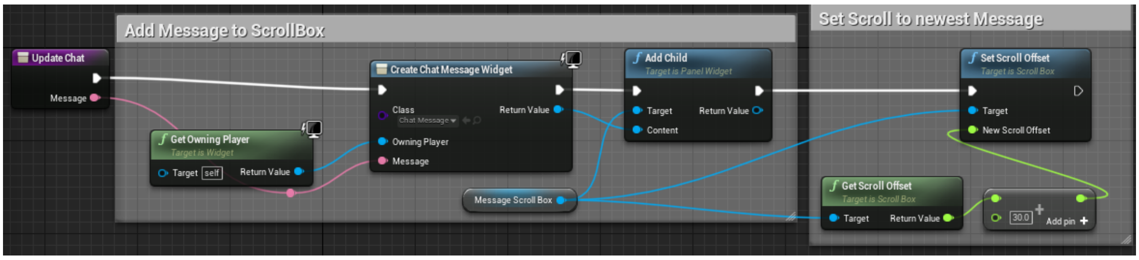
Picture 14. Small modification to the end of keyboard blueprint.



Picture 15. Clicking the Send -button functionality part 1.

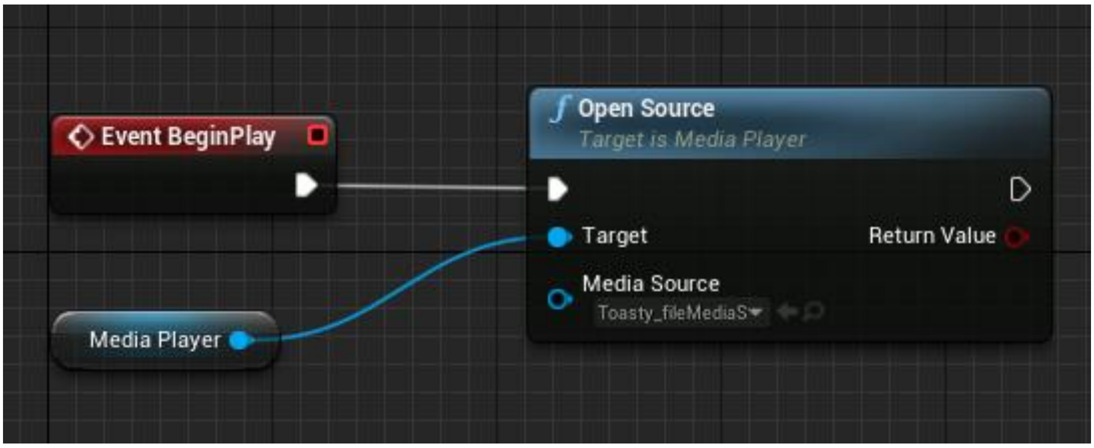


Picture 16. Clicking the Send -button functionality part 2.



Picture 17. Update chat -function.

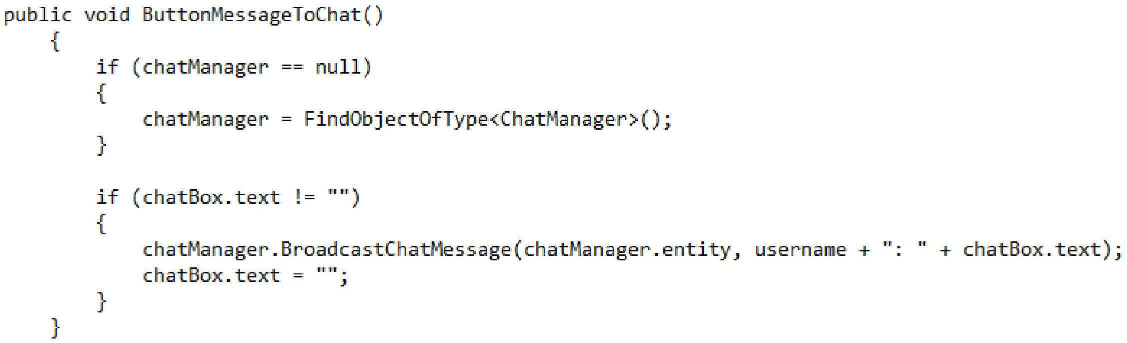
Adding a video player to the application was straightforward as there was a tutorial from Epic games that explained how to add a media player and a texture to display the video on a 3D object. A 3D canvas with a video player attached was added to the virtual environment and it required no extra work to implement multiplayer functionality as the video was displayed to all users similarly by default.



Picture 18. Media player blueprint.

Unity

A text chat was implemented by adding a local input field for every user and sending the typed string to the chat window that is broadcasted to all the users over network. The input field has a script that adds the username in front of the typed text and calls the ChatManager function BroadcastChatMessage once the user presses the send button in the UI.

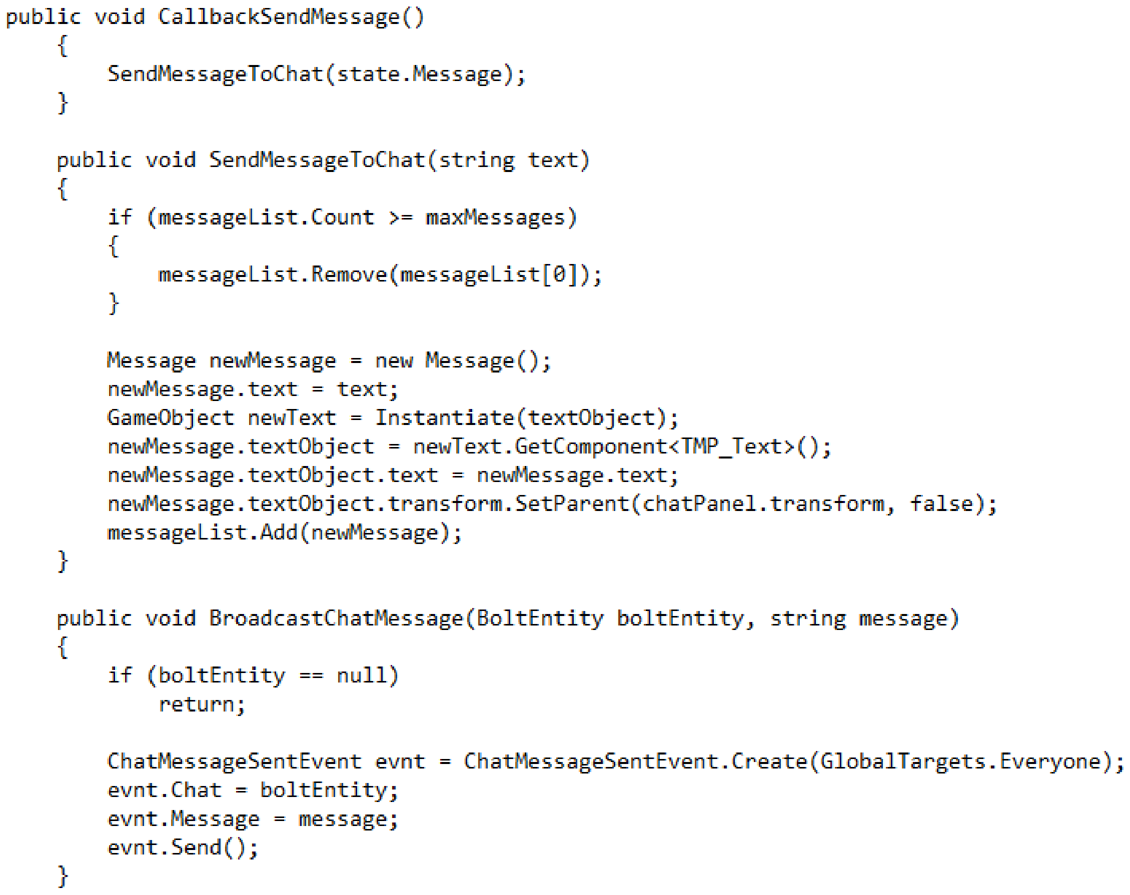


```
public void ButtonMessageToChat()
    {
        if (chatManager == null)
        {
            chatManager = FindObjectOfType<ChatManager>();
        }

        if (chatBox.text != "")
        {
            chatManager.BroadcastChatMessage(chatManager.entity, username + ": " + chatBox.text);
            chatBox.text = "";
        }
    }
```

Picture 19. ChatInput -functionality.

ChatManager then creates a new ChatMessageSent event that updates the message string variable for all the users which triggers a callback. The callback calls a function that updates the chat window with the message that was sent by the user.



```
public void CallbackSendMessage()
    {
        SendMessageToChat(state.Message);
    }

    public void SendMessageToChat(string text)
    {
        if (messageList.Count >= maxMessages)
        {
            messageList.Remove(messageList[0]);
        }

        Message newMessage = new Message();
        newMessage.text = text;
        GameObject newText = Instantiate(textObject);
        newMessage.textObject = newText.GetComponent<TMP_Text>();
        newMessage.textObject.text = newMessage.text;
        newMessage.textObject.transform.SetParent(chatPanel.transform, false);
        messageList.Add(newMessage);
    }

    public void BroadcastChatMessage(BoltEntity boltEntity, string message)
    {
        if (boltEntity == null)
            return;

        ChatMessageSentEvent evnt = ChatMessageSentEvent.Create(GlobalTargets.Everyone);
        evnt.Chat = boltEntity;
        evnt.Message = message;
        evnt.Send();
    }
```

Picture 20. ChatManager functionality.

A video player was as easily added in Unity as in UE4. A render texture was created and a video file was dragged in to the video player. After that, a 3D plane was created and the material was set to be a render texture. Interestingly the video started from the beginning for each user once they joined the session.

4.5 Further development

Due to lack of time some features were not implemented and were left to design stage. There were some ideas on how the features could be quickly implemented if the testing should be continued after the completion of this thesis.

4.5.1 Privacy options

Unreal Engine 4

The text messaging plugin Complete Chat System had a functionality for private channels and would be the easiest and fastest way of implementing privacy into the text messaging. (Unreal Marketplace 2021)

Opening a private voice chat between two users is not something that can be easily found in a tutorial or a guide. One way of implementing it could be accessing an individual user within the text chat and opening a channel through there or accessing a user through a 3D line trace in a virtual space and connecting to their audio listener. These solutions are purely theoretical at this point and no tests have been done if either of these would be a viable solution.

Unity

Photon Bolt’s event-system offers the possibility to specify which users get the information delivered to them. This could be utilized to create a private chat system, but the challenge would be determining the specific user that gets the message.

A faster way would be to use Photon Chat plugin that offers different channel options for the chat and it is also included in the Photon Cloud subscription. (Photon 2021)

4.5.2 Screensharing

Unreal Engine 4

There were not any tutorials or guides available on how to achieve screen sharing with UE4, but a plugin called Screen capture media source can convert a desktop view into a media source. This means that the desktop view could be handled in a similar fashion as any normal video feed. The plugin is available for purchase in Unreal Marketplace. (Unreal Marketplace 2021)

Unity

4.5.3 Animations and show cases

Unreal Engine 4

Animations are something that UE4 handles very well, and the engine is even used in film industry for this purpose (Unreal Engine 2021). There is a built-in tool called Sequencer that can be used to place different variables on a timeline. It can be used to place a certain object in a certain position in a specific moment and then to another position in a later moment. The Sequencer then interpolates the movement between these moments and creates an animation where the object moves in a steady pace between the positions during the given timeframe. This tool could be used to create an animation of the desired scenario that demonstrates whatever situation needs to be show cased.

Unity

Unity has a similar system for animations that UE4 has and it is called Cinemachine. It follows the same principles as the UE4 Sequencer.

4.5.4 Augmented Reality

Unreal Engine 4

Varjo template had a working example of AR-element implemented in an UE4 level which could be utilized in application development (Varjo 2021). It basically sets a material in an object that acts as a mask for the camera in the Varjo XR-1 headset and projects the camera’s output in the material creating an experience of augmented reality. This could be utilized in a way that only other users and a certain part of the current environment would be virtual, and rest of the environment would be the real world that is around the user. This would create the experience of having a conference or a product presentation at the users living room.

Unity

Varjo provides template for Unity with a working AR-element as well. It works pretty much the same way as in UE4.

4.5.5 Artificial intelligence

Unreal Engine 4

Artificial intelligence or AI would be used in the environment as a text messaging bot that would be available for user feedback and help. The simplest way of accomplishing this would be to rely on a plugin. ChatBot A.I. Simulator is a plugin for UE4 that brings a bot in to the chat that can answer some prespecified questions. It is not a true AI in a sense that it does not learn new things but rather simulates a human interaction in a text messaging platform. In theory it could be combined with a machine learning application to make it adapt as it gets more repetition, but this is only speculation at this point.

Unity

A plugin would be the simplest way to handle a text messaging bot in Unity and one suitable candidate could be Chatbot by Nexus Gamesoft. This one uses actual AI instead of simulating human interaction.

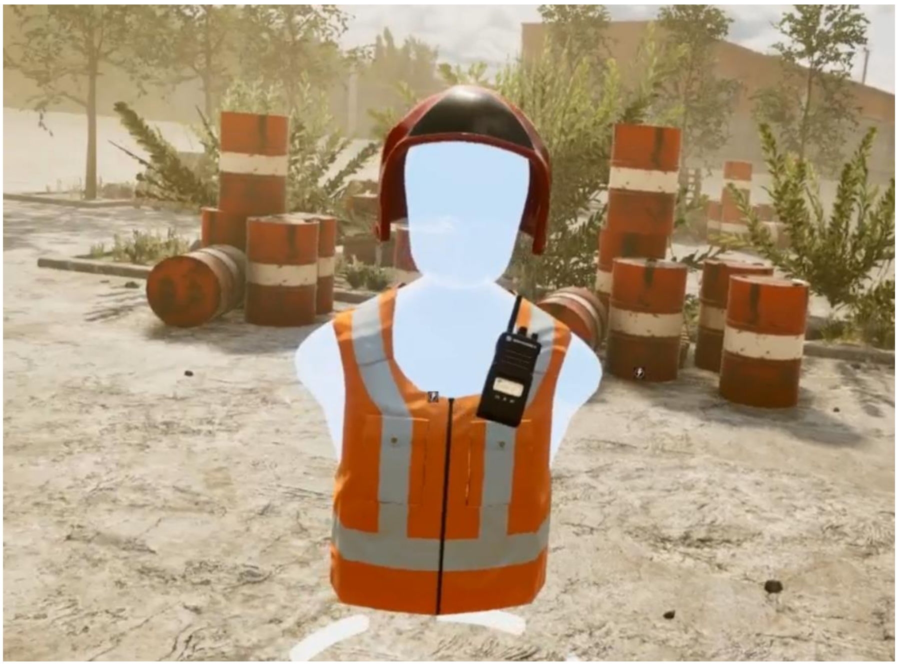
5 TESTING

The testing was done by a small group of expert testers with very little guidance beforehand and with no supervision. The testers were provided with two different applications. One made with Unity and the other with UE4. It should be noted that the purpose of the testing was not to find things that were broken but rather find out what would be a good approach to implement the specific features and are the expert testers able to use the applications without guidance. As such the parts that did not function correctly or had issues are not covered by this study.

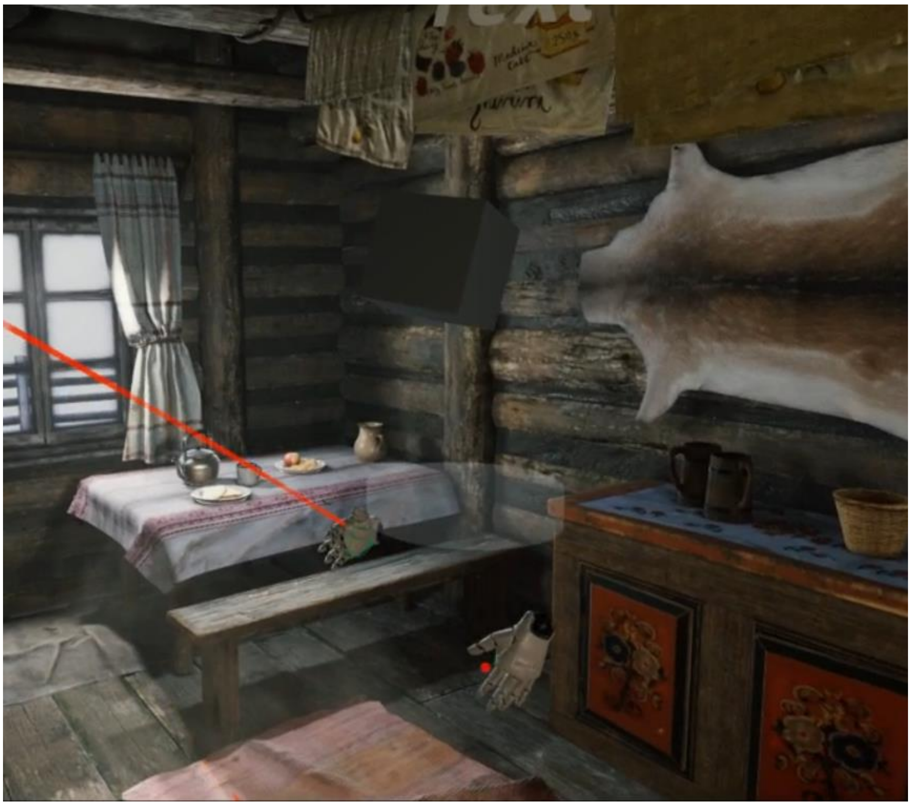
5.1 Unity and Unreal application differences

The applications had some differences in the features to see which were preferred by the testers. Unity version had a text chat that used hardware keyboard inputs and the UE4 version had a virtual keyboard. The input for UI interaction in Unity was the A-button in Oculus Touch controller and in UE4 the same interaction was handled with the Trigger button. VoIP had a channel open all the time in Unity and UE4 version had the channel open only while holding down the A button. The UE4 version had a more sensitive inputs with teleportation that triggered when the user touches the joystick while Unity required a press of the joystick. Video player functioned in a similar way on both applications.

Each application also had different environments. Unity version had a very basic scene that was made for Sami Laukkanen's previous project and the Log Cabin by Gabro Media asset was used as an environment for UE4 version. The latter had ambient sounds in the environment while the Unity environment was completely silent. Unity had 3D models for user avatars and UE4 version had only cubes to display user position and rotation.



Picture 21. Unity avatar and environment.



Picture 22. Unreal Engine 4 avatar and environment.

5.2 Test results

Unity

The text chat in the Unity version made the testers request a virtual keyboard immediately after they realized they needed the hardware keyboard to type. Some testers never realized how the chat was supposed to work. Trying to type with a hardware keyboard appeared to be difficult with VR headset. Testers also noted that the chat should be something that travels with the user instead of being tied to a certain location inside the environment.

There also appeared to be a small lag with the UI buttons on client side and the testers hoped that there would have been some sort of feedback when button was pressed.

The usage of A-button for UI interactions felt more natural to some testers and a suggestion of using both A-button and the trigger for UI interaction could be a good solution.

Unity’s teleportation felt good according to testers as they had to really press the button before they teleport.

The testers felt that with a little polishing the application has a lot of potential as a meeting tool, but it would still require a possibility to access the application without a VR headset and customization options. An introduction or a tutorial at the beginning was suggested to get the user more acquainted with the application.

Unreal Engine 4

Text chat with a virtual keyboard seemed to be a better solution than using hardware keyboard. The testers noticed that although they were unable to see each other’s output while typing they were able to see which button highlights on the keyboard. That highlighting should be removed according to testers.

The UI buttons worked better in UE4 than in Unity and there was no noticeable lag on client side.

Some testers found it hard to remember to hold down A-button while communicating and some voice messages were left unsent because of this.

The teleportation inputs seemed to be too sensitive and testers teleported unintentionally on some occasions which they found annoying. They suggested that the inputs should be more like in the Unity version.

The testers felt that the environment in UE4 version created a nice immersion with the lighting and the ambient sounds, but all agreed that the ambient sound effects were too loud outside the cabin. The volume was much better inside and did not get in the way of VoIP communication.

The general impression on the UE4 version was even better than the Unity version but the testers agreed that it would take TUAS a lot more effort to create and polish such an application with Unreal Engine.

5.3 Analysis and the Outcome

Expert testing clearly revealed some usability issues with both versions which should be addressed if a more polished application would be made. Most of these were small things that could make a huge difference on the user experience.

The text chat should have a virtual keyboard attached to it as typing with a hardware keyboard is very difficult in VR. On the other hand, there should be a possibility to use hardware keyboard if the user accesses the application without the VR headset.

Unity version had an issue with a lag when interacting with UI elements which should be fixed with lag compensation. UI interactions and their input bindings require some more research to find an optimal solution. The tests gave no clear indication which input button was preferred for these actions.

The VoIP system could be altered in a way that a certain input opens the voice channel and leaves it open until next input. This solution would not require the constant holding down of the input button and remembering to do so.

Teleportation needs to be triggered only after the button is pressed and not if only touched. It should be noted that all the testers had previous experience of a teleportation system where input needs to be pressed and none had experienced a different approach before.

A tutorial or some introduction at the start would help the users with little previous experience in similar applications as even all the expert testers were not able to use all the features without external guidance. A short guidance at the beginning that tells the user what can be done inside the application and how would probably be sufficient.

The environment and ambient sounds seemed to create a better immersion and is something that should not be overlooked if the application would be continued further.

It was encouraging to notice that although both of the applications were developed by a single developer in less than a month most of the required features worked in an acceptable level and could be used as a base on which to build the virtual conference application TUAS School of ICT requested.

6 CONCLUSION

The objective of this thesis was to determine whether it would be feasible to develop a custom virtual conference application or simply use the existing applications. A list of requirements provided by TUAS School of ICT was used as a benchmarking tool for the comparison. An additional objective was to determine if UE4 would be a suitable candidate for the development of such an application.

The thesis also examined different VR conference platforms currently available at the market, what features they contain, and what differences exist between them. The intention was to focus more on the aspects that TUAS School of ICT felt more important for the platform and less on the platforms that were intended for more recreational purposes. Surprisingly, this overview of different platforms showed that despite the huge budgets that Microsoft and Facebook have invested in their applications their products were not that far ahead of their smaller competitors and one could argue that Facebook is still behind most of them with Horizon still being in a closed beta-testing. All in all, the existing VR platforms are very similar and there are not immense differences between them.

The TUAS Virtual Conference Center case study had many challenges due to the huge scope of the project. The initial plan was to develop a prototype of the application, but that plan failed quickly as the development problems with UE4, multiplayer and VR combination proved out to be more complex than they first appeared. These problems caused the study to trim out many features that were originally requested.

The thesis could have been improved by comparing more frameworks and game engines before initiating development and making a more enlightened choice on the matter. UE4 proved out to be a good environment to test the requested features because of the wide variety of plugins available and the reliability of the engine. The only downside was that using the engine is slow if the developer is not familiar with all the features it contains and learning all of those could take a great amount of time.

The other version of the application was made with Unity and it took a lot less time since it was more familiar environment to work with. It should be noted that previous experience with Unity was approximately tenfold compared to experience in Unreal. The two small prototypes were achieved in a timeframe of less than a month by a single developer, but the outcome lacks polishing and is not a beginner friendly experience.

If the projects would have been done again from the beginning it would have been faster to create the project only on Unity as more experts and experience would have been available to help with the many complex issues that multiplayer development has. On the other hand, it might be easier to develop a fully polished version of the application on UE4 now with more experience with the engine as UE4 forces the developer to a certain software architecture that keeps larger projects more manageable.

The most innovative part of this study is most likely the development of VR multiplayer application features with UE4 as there does not exist much information regarding it. The VR Expansion plugin gives an excellent starting point but adding custom features on top of it was a challenge without guides or tutorials easily available.

The answer to the initial question of whether it would be more feasible to develop a custom VR platform is not a definite no but if such a platform would be developed it would take more time and resources the more features it should include. It should be noted that even with generous financial resources, the development of this kind of application is not easy as can be seen from the case of Facebook’s Horizon. Success in development seems to be more dependent on a good design and the right tools for the task. According to the experts, in TUAS a prototype could be achieved relatively quickly but a polished version could take several years in a worst-case scenario if it were to have the same number of features that the existing platforms have.

Both projects could be developed further and the first thing to do would be to take notes from the expert testers recommendations on already implemented features and do the necessary fixes. After those are completed, the next step should be stress testing the networking to find out how many users it would be possible to have in the same room and adjusting the networking and features based on the results. Finally more of the requested features could be implemented and constantly tested to make sure the networking can handle the new features.

REFERENCES

Constine, J. 2019. Facebook announces Horizon, a VR massive-multiplayer world. Referenced 28.4.2021 Available at : https://tcrn.ch/2nbG9De

Helsinki XR Center 2021. Welcome to Helsinki XR Center – the Home of Extended Reality! Referenced 10.6.2021. Available at :- https://helsinkixrcenter.com/

Helsinki XR Center 2020a. HXRC Team: Testing AltspaceVR, a social VR platform. Referenced 28.4.2021 Available at : https://helsinkixrcenter.com/hxrc-team-testing-altspacevr-a-social-vr-platform/

Helsinki XR Center 2020b. HXRC Team: Testing Glue, a virtual collaboration tool. Referenced 24.4.2021 Available at : https://helsinkixrcenter.com/hxrc-team-testing-glue-a-virtual-collaboration-tool/

Helsinki XR Center 2020c. HXRC Team: Testing Mozilla Hubs, a mobile-friendly VR platform. Referenced 28.4.2021 Available at : https://helsinkixrcenter.com/hxrc-team-testing-mozilla-hubs-a-mobile-friendly-vr-platform/

Helsinki XR Center 2020d. HXRC Team: Testing VRChat, a community creation focused social VR application. Referenced 28.4.2021 Available at : https://helsinkixrcenter.com/hxrc-team-testing-vrchat-a-community-creation-focused-social-vr-application/

Järvinen, N. 2019. Implementation of virtual reality collaboration platform with Glue. Bachelor's thesis. Häme University of Applied Sciences. Referenced 21.5.2021 Available at: www.theseus.fi

Lang, B. 2020. Preview: Facebook 'Horizon' Aims for a Sweet Spot Between 'Rec Room' & 'VRChat'. Referenced 28.4.2021 Available at : https://www.roadtovr.com/facebook-horizon-beta-preview-hands-on/

Matt. 2019. CouchLearn: The easy way to add Voice Chat into your multiplayer Unreal Engine 4 Game. Referenced 28.4.2021 Available at : https://couchlearn.com/the-easy-way-to-add-voice-chat-into-your-multiplayer-unreal-engine-4-game/

Matthews, B; See, Z; Day, J. 2020. Crisis and extended realities: remote presence in the time of COVID-19. Media international Australia incorporating Culture & policy 2021-02, Vol.178 (1), p.198-209

Microsoft. 2021a. AltspaceVR Documentation: Creating an event. Referenced 29.4.2021 Available at : https://docs.microsoft.com/en-us/windows/mixed-reality/altspace-vr/tutorials/creating-an-event

Microsoft. 2021b. AltspaceVR Documentation: Roles in events, worlds, groups, and in AltspaceVR. Referenced 29.4.2021 Available at : https://docs.microsoft.com/en-us/windows/mixed-reality/altspace-vr/getting-started/roles

Photon Engine. 2021. We Make Multiplayer Simple. Referenced 28.4.2021 Available at : https://www.photonengine.com/en-US/Photon

Roy, A. 2021a. AltspaceVR Review: Microsoft's Social VR Offering. Referenced 28.4.2021 Available at : https://www.xrtoday.com/virtual-reality/altspacevr-review-microsofts-social-vr-offering/

Roy, A. 2021b. Glue Review: A New Era of Business Comms. Referenced 28.4.2021 Available at : https://www.xrtoday.com/reviews/vr-reviews/glue-review-a-new-era-of-business-comms/

Roy, A. 2021c. Mozilla Hubs Review: Open Social VR Collaboration. Referenced 28.4.2021 Available at : https://www.xrtoday.com/virtual-reality/mozilla-hubs-review/

Statzer, J. 2021. VR Expansion Plugin, A Virtual Reality Toolkit. Referenced 28.4.2021 Available at : https://vreue4.com/

Stupid Industries. 2020. Unreal Marketplace: Screen capture media source. Referenced 28.4.2021 Available at : https://docs.unrealengine.com/marketplace/en-US/product/38b887ee46ef471ebdb93d8cfb2ad96a

Unreal Engine. 2019. Unreal Engine Forum: Best peer to peer network solution for a multiplayer shooter. Referenced 28.4.2021 Available at : https://forums.unrealengine.com/t/best-peer-to-peer-network-solution-for-a-multiplayer-shooter/129270

Varjo. 2021. Developer portal: Downloads. Referenced 28.4.2021 Available at : https://developer.varjo.com/downloads

Volpe, J. 2020. The weird and wonderful world of VRChat is a joyful antidote to these dark times. Referenced 28.4.2021 Available at : https://mashable.com/article/vrchat-antidote-dark-times/?europe=true

Interview questions

Interview questions about AltSpace VR for Jami Aho and Panos Markopoulos:

- Tell a little bit about yourself and your experience in game development
- What did you achieve with AltSpace?
- What was the hardest part to implement?
- Was there something you wanted to do but couldn’t because of application limitations?
- Did you have any trouble with big audiences in AltSpace? How many users were there during the peak?
- Is it possible to create separate rooms within the scene?
- Is it easy to add videos and photos to the scene?
- Is there a possibility to create a 360 photosphere?
- Is there a private meeting or voice channel possibility?
- Are video calls possible inside the app?
- Is it possible to edit skybox and lighting?
- Is there a built in minimap?
- Is it possible to have access control inside the app?
- Is it possible to add bots or feedback systems?
- Is it possible to mod and add features to the app?

Interview questions for Sami Laukkanen:

- Tell a little bit about yourself and your experience in game development
- Describe the project you made shortly?
- Did you use some plugins to create the project?
- What would you consider to be the user cap with the tools you had?
- What would it require to raise that cap?
- How long would it take you to create a similar app as AltSpace by yourself?
- How long would it take if you had all the resources you could wish for?
- What would be the hardest thing to implement and why?

Interview results

Jami: Graphics, started with 2D in own company, using Unity. Worked as assistant while studying, done Moomin games with unity for few years. Now working for FIT-research group in TUAS as a project engineer.

Panos: Working as game designer, world building for Altspace, bachelors in game design in London, masters in VR in London.

Panos: We created the scene in Unity and imported it to Altspace. It's a process but fairly easy once you learn it. It would take a lot longer to create a similar multiplayer social experience from scratch.

Jami: pointed out video importing problems

Panos: Problems with how video players work in AltSpace. Altspace was updated during the work. Too many videos proved out to be problematic.

Jami: A lot of videos take a lot of time. Lot of manual work with quality adjusting.

Panos: Loading too many videos from external sources clogged the whole application.

Panos: Only way to implement scripting is by uploading SDKs which need to be approved by Altspace.

Jami: There are approved SDKs that can be utilized. One of those was the video player.

Panos: Altspace allows 50 people in one room, we had about 80 at most. A speaker can be mirrored for a big audience but they exist in different rooms. For example 200 people could see the speaker but all people would not see everyone, just the ones in their room. Video starts from the beginning for each client so it is not in sync with other users.

Jami: photos were a lot easier.

Panos: photos are easier and can be applied with a json file from external source

Panos: we have four 360 photospheres inside the space.

Jami: basically It was just a 3D sphere with inverted materials.

Panos: best way to go about with privacy would be to create private rooms that only specific users have access to. No private voice chat but text messages can be sent privately. User names need to be added to access point by the host. Private meetings with new contacts is possible but not very convenient.

Panos: Sharing a screen requires admin rights or a host needs to run the video call between two clients. Easy for the host, not so easy for the clients.

Panos: Altspace doesn’t provide any bots. Could be achieved by creating a custom SDK but not very easy.

Panos; It would be possible to create a Altspace copy but if it needs to be very polished, it could take years and we would need servers to run it. It would be a very big project but possible. No two guys create altspace.

Jami: It needs a team and a lot of money.

Panos: Glue seems to be the best one I know. Can make your own scripts, need to be ran through Glue. More features and better looking visuals. All users need account and paid subscription. Same freedom of editing as Altspace and more. Direct improvement compared to Altspace. Huge problem with subscription model. Gets very expensive.

Panos: If we are to make something like this it should be started ASAP because it would take a lot of time and money.

Jami: There is so many extra things to consider when making a multiplayer application.

Panos: There should be modification tools available too and they have huge risks.

Interview with Sami

Sami: Project engineer in TUAS, student assistant first during studies. 3,5 years as engineer, programmer, mostly VR with Unity, some multiplayer

All the programming except plugins, parking lot with burning barrel and extinguishing equipment, teacher as a host, students join as clients and extinguish fire after teacher shows what to do.

User cap 32 or 64 can’t remember. Tested with 1 host and 2 clients. No stress testing done, project was proof of concept.

Photon Bolt was used for networking, Photon Voice was used for proximity voice chat. Steam VR was also included in the project for basic VR functionality. Mirror was not tested for this project.

Users in one scene is limited to 64 by photon bolt. Paid subscription allows 5000 simultaneous users in a server but not in a same scene. Bolt seems to be better with smaller user amounts and Photon Pun would work better with larger crowds.

If a similar application as AltSpace would be made an important thing to consider would be to allow users to create their own environments. AltSpace does this by using scenes made in Unity in a package that includes all the assets in a scene.

Altspace is fairly simple with it’s multiplayer elements as you can’t affect the world too much. Admin side of things requires more effort. Hardest part would be the backend side of things.

If done alone it would take me year or two mostly because of learning the backend side of things. With a good team it would probably take four months to create a working prototype. It would require a real good backend developer. If all required developers from the lab would be employed it would take about four months. Polished version would take about a year with those resources. Since a similar project has not been made yet, there might be some unforeseen aspects that could delay the process.

General security of the connection and accounts would be a major concern if the product would be commercialized.