Richtlinie zum „Software-Sprint“

XRevent_0.2 – Broadcaster

Schlussbericht

Zuwendungsempfänger:

von der Heide, Fischer, Ruppel & Milter GbR
Anke von der Heide, Dr. Patrick Tobias, Thorbjörn Ruppel, Tom Milter

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS22S25 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

In den vergangen drei Jahren der Corona-Pandemie haben wir weltweit gelernt die Vorteile von hybriden Events zu nutzen. Für Besucher*innen waren sie während der Pandemie die einzige Möglichkeit an wissenschaftlichen und kulturellen Veranstaltungen teilzunehmen und für die Kultur- und Eventbranche, immerhin Deutschlands sechst größtem Wirtschaftszweig, die einzige Möglichkeit Einkommen zu sichern. Zahlreiche Untersuchungen zeigen, dass auch nach der Pandemie digitaler Veranstaltungen viele Vorteile mitbringen.

Einer der Gründe ist, dass viele Besucher*innen von Konferenzen, Messen, Festivals, Theater und Clubs die Vorteile digitaler bzw. hybrider Darbietungen kennenlernten und sie auch heute noch nutzen. Für Veranstalter*innen lassen sich neue Zielgruppen und eine größere Reichweite erzielen, nicht nur um die Ausfälle der letzten Jahre zu kompensieren. So sieht es auch eine Studie zur Buchungslage für Veranstaltungen bis 2024. Nach Einschätzungen der Veranstalter*innen und Anbieter*innen werden die Buchungen für virtuelle und hybride Veranstaltungen sogar höher sein als für Präsenzveranstaltungen. So ergibt sich eine Steigerung im Jahr 2024 von 103% für Präsenz, 128% hybride und 120% virtuelle Veranstaltungen gegenüber 2019.[1]

Allerdings hat nicht jede Künstler*in die Fähigkeit oder Möglichkeit diese Darbietung kostenneutral und ohne Programmieraufwand zu erstellen. Zu diesem Zweck wurde der XRevent Creator in der Prototyp Fund Runde 10 entwickelt, ein Open-Source Baukasten für Cross-Reality Events, der es ermöglicht einfach virtuelle Veranstaltungsräume zu gestalten, die angehängt an die XRevent Plattform für Besucher zugänglich zu machen und über einen Ticketshop zu monetarisieren.

[1] German Convention Bureau. (2022). *Wie schätzen Sie die Buchungslage für Veranstaltungen für die nächsten drei Jahre ein?*. *Statista*. Statista GmbH. Zugriff: 28. März 2023. https://de.statista.com/statistik/daten/studie/1246847/umfrage/umfrage-zur-einschaetzung-der-buchungslage-von-veranstaltungen-nach-corona/?locale=de

Wie viele andere XR Plattformen oder Metaversen, konnte auch XRevent Live-Inhalte nicht latenzfrei in die virtuelle Welt streamen. Zumeist behelfen sich andere Plattformen mit vorab aufgezeichneten Videos, deren Stream zu einem bestimmten Zeitpunkt live gestartet wird. Für eine Cross-Reality Plattform ist es unsere Meinung nach unerlässlich, dass beide Veranstaltungen, reale wie virtuelle, gleichzeitig Live stattfinden. Dies ermöglicht nicht nur dass der Performer durch ein Live-Feedback in den realen Raum ein Gefühl für das Publikum im Virtuellen bekommen, sondern auch dass die Besucher*innen beider Veranstaltungen miteinander interagieren können und wissen was in dem jeweils anderen Veranstaltungsraum los ist.

Zudem ist der Anspruch unabhängiger von teuren Assets und Software zu werden. Für die Unity Anwendung von XRevent wurde das Asset AVPro genutzt. Dies kostete in unserer Version von 2021 jährlich 400€ und heute 750€. Existierende freie Tools bieten nicht den nötigen Umfang oder sind Lösungen die Firmen wie NewTeck an kostenpflichtige Hardware binden. Bei andere Anbieter ist die Software kostenfrei, allerdings nicht die Streamingleistung pro Minute. Diese kann sich nicht jeder leisten und bietet trotzdem noch kein latenzfreies Streaming.

In der PT Runde 12 wurde deshalb der XRevent Broadcaster entwickelt. Er ermöglicht es Videos mehrere Kameras über einen Bildmischer live zu streamen. Er soll vor allem zwei Aufgaben erfüllen. Die hohen Latenzen von bis zu 30 Sekunden des Livestreamings auf unter eine Sekunde zu reduzieren und somit einen Feedback Loop in den realen Veranstaltungsort zu ermöglichen und die Aufnahme von Videos für Dokumentationen oder Archive zuzulassen.

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Da die fünf Punkte Community, Social, Flexibel, Barrierfrei und Datensicherheit weiterhin die wichtigsten Aspekte des Projekts XRevent sind, sollen sie auch in diesem Schlussbericht die Struktur dieses Absatz bilden. Sie wurden daher um neugewonnene Ziele beziehungsweise Ergebnisse des Projektes ergänzt.

Community

Dem Wunsch eine XRevent Commuity zu bilden sind wir mit Hilfe eines Community Coachings näher gekommen. Vor allem konnten wir uns mit bestehend Communities und Branchenverbänden wie: XRHub Bavaria, Next Media Hamburg, Next Reality Hamburg, Rock City Hamburg, Gegenlicht, Scenografen Bund vernetzen. Das Feedback dieser Verbände ist durchweg positiv und schon jetzt erhalten wir viel Unterstützung bei der Umsetzung von XRevent.

Social

Das Ziel ist es eine Teilhabe vieler an Kultur- und Wissensveranstaltungen zu ermöglichen und das zu einem erschwinglichen Preis. Wir glauben daran, dass sich erweiterte SocialVR und der Vernetzungsgedanke von realen Events nicht ausschließen. Deshalb liegt die Priorität von XRevent darin latenzfreie Signalwege zwischen der realen und virtuellen Welt zu meistern. Das entspricht auch dem Wunsch einer Vielzahl von Befragten (44%) in einer McKinsey Studie aus dem Jahr 2022.[2] Die geben an, dass sie an der Teilnahme am Metaversum vor allem die Vernetzung mit anderen Menschen schätzen.
Aus diesem Grund ist die XRevent eine Multiuser Plattform. Besucher*innen können miteinander interagieren und mittels Voice-Chat sprechen. Darüber hinaus ist die reale Veranstaltung mit der virtuellen Veranstaltung durch Interaktionen zwischen Besucher*innen aller Räume verbunden. Das

[2] McKinsey. (2022). *Was reizt Sie am meisten an der Teilnahme am Metaversum?*. *Statista*. Statista GmbH. Zugriff: 29. März 2023. https://de.statista.com/statistik/daten/studie/1321928/umfrage/gruende-fuer-die-bevorzugung-virtueller-aktivitaeten-im-metaverse/?locale=de

kann über interaktive Installationen die im real wie virtuellen Raum stehen geschehen oder mittels Anruf über das Handy in die virtuelle Welt.

Flexibel

Durch die Integrierung des XRevent Broadcasters in das Webeinterface des XRevent Creator wird dieser noch flexibel einsetzbar. Mit wenigen Klicks lassen sich entweder interaktive Video-Inhalte aus TouchDesigner oder Videomappings aus Resolume einfügen. Aber auch Kamerabilder mehrere Video- oder Handykameras in die VR Räume des Creators integrieren. Somit wird es den Nutzer*innen erleichtert die eigene vorhandene Hard- oder Software zu nutzen, ohne zusätzliche Technik ausleihen oder gar kaufen zu müssen.

Barrierefreiheit

Der Performer kann ganz einfach die Darbietungen und Reaktionen des Publikums live aus dem VR Raum verfolgen. Dem Anspruch Applikationen für möglichst viele Endgeräte zur Verfügung zu stellen, damit Besucher*innen möglichst barrierefreien Zugang erhalten, wurden der Broadcaster noch nicht 100% gerecht. Bisher funktioniert er nur für Android und somit die VR-Brillen Oculus Quest 2 und Pico 4. Updates für den PC Mac und Linux sollen bald folgen.

Datensicherheit

In dieser Förderphase haben wir uns besonders mit dezentralen und föderierten Multiversesystem auseinandergesetzt, die offene Standards nutzen, um Datenmissbrauch und dem Diktat globaler Unternehmen zu entfliehen. In einem nächsten Schritt sollen deshalb Profilseiten entstehen die Nutzerdaten nur dezentral speichern. Ein weiterer Schritt in diese Richtung wäre ein Open Source Multiplayer der Positionsdaten ebenfalls nur dezentral zu verarbeiten.

Ausführliche Darstellung der Ergebnisse

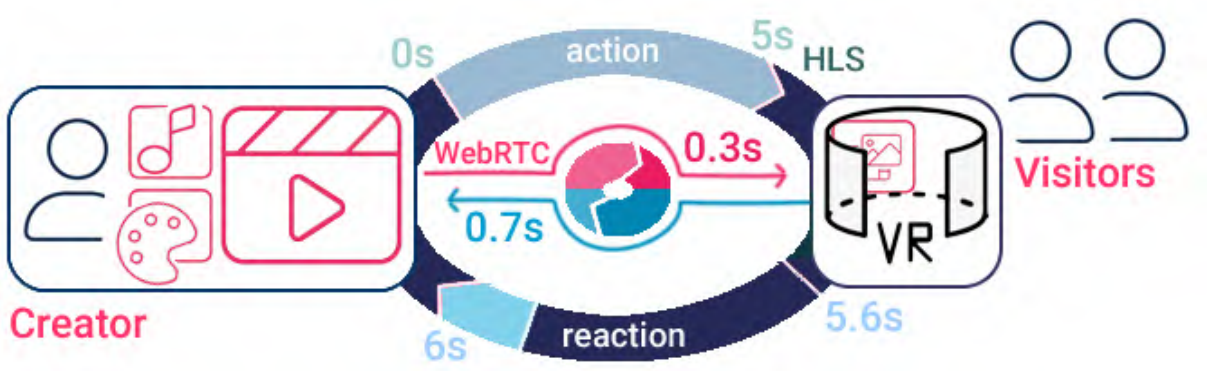
Die finale technische Umsetzung umfasst im wesentlichen drei Unterprojekte die es zusammen erlauben latenzreduziertes Video in eine mit Unity erstellte virtuelle Welt zu streamen. Den Kern bildet das (1) Android-Plugin welches vorkompiliert als Android Archiv (AAR) in Unity Projekte importiert werden kann und das eigentliche Videorendering übernimmt. Die (2) Unity XRevent-Broadcaster App zeigt die eigentliche Anwendung des Plugins, indem es Anfragen an das Plugin richtet und auf dessen Nachrichten reagiert. Um einen Videostream an mehrere Clients zu verteilen, benötigt es eine Streaming Infrastruktur mit entsprechenden „Videoeingängen“ und „-ausgängen“. Hier implementiert der XRevent-Broadcaster einen sog. (3) „Provider“ und „Subscriber“ um einerseits Video per Browser entgegen zu nehmen, aber auch auszuspielen. Dieses Unterprojekt wurde mittels React realisiert.

1	Android Plugin	https://github.com/xrchisense/xrevent-broadcaster-android
2	Unity Application	https://github.com/xrchisense/xrevent-broadcaster-unity
3	Web Application	https://github.com/xrchisense/xrevent-broadcaster-react

Allgmeines zum Streaming

Streamingtechnologien sind grade im Zuge von COVID-19 allgegenwärtig in unserem Arbeits- und Privatleben. Jedoch gibt es nur wenige Standards, auf denen Applikationen wie YouTube, Zoom, BigBlueButton, Jitsy, etc. aufsetzen. Im Wesentlichen reduzierte sich nach einer Analyse verschiedener Netzwerkprotokollen die Möglichkeiten auf zwei zeitgenössische Möglichkeiten: HTTP Live Streaming (HLS) so wie es YouTube und Co. verwenden oder WebRTC (Web Real-Time Communication). Beide Standards sind in vergleichsweise jung, wenn man ihre Technologie in Bezug zur heutigen Marktdurchdringung setzt. Die Erstveröffentlichung von HLS war im Mai 2009 mit den RFC8216 und WebRTC wurde 2011 erstveröffentlicht mit einer stabilen 1.0 Version im Mai 2018.

Ohne zu sehr in die Details der beiden Netzwerkprotokolle einzusteigen, möchten wir hier bloß kurz den Entscheidungsgrund nennen, weshalb wir uns für WebRTC als Kerntechnologie entschieden haben. Bei Protokolle unterstützen den (Web) Video Codec Standard H.264 und werden wohl auch zukünftig mit H.265 und VP9 gemeinsame Wege in Bezug auf Videokompression gehen. Jedoch unterscheiden sich die Auslieferungsstrategien der Protokolle. Die Grundstrategie von HLS ist Videos zu fragmentieren und diese per HTTP-Server auszuliefern. Dies hat den Vorteil einer extrem hohen Skalierbarkeit in Bezug auf die Anzahl der Konsumenten aber auch der Qualität des Streams, da HTTP-Delivery heutzutage sehr leistungsstark und elastisch ist. Die kleinen Teilstücke des Videos, die sogenannten Segmente können schnell auf weiteren Servern gespiegelt werden und Videos können für verschiedene Endgeräte in verschiedene Auflösungen abgelegt werden. Dies hilft unter anderem dabei einen unterbrechungsfreien Stream herzustellen bei schwankender Bandbreite. Den immensen Vorteil, den die Segmentierung für die Skalierbarkeit bringt, ist jedoch auch gleichzeitig sein größter Schwachpunkt für unseren Anwendungsfall, denn ein Segment hat eine vorgeschriebene Minimallänge von ca. 6 Sekunden. Insgesamt geht man von einer Länge zwischen 4-12 Sekunden aus um eine gute Balance zwischen Videoqualität und Netzwerkoverhead zu schaffen. Auch wenn es hier Weiterentwicklungen gibt die Low-Latency HLS (LL-HLS) oder LL-Dash, welche Segmente in weitere Parts unterteilen, ist die theoretische Latenz immer noch 2-5 Sekunden, während sie bei WebRTC bei ca. 500 Millisekunden liegt.



Um einen im Live-Show relevanten Kommunikationskreislauf zwischen Performer und Zuschauer herzustellen und aufrecht zu erhalten, sind bereits 2 Sekunden zu viel, weshalb wir und für WebRTC entschieden. Der Schwäche die WebRTC jedoch hat, ist wiederum die vorhin beschriebene Stärke der Skalierbarkeit von HLS. Denken wir z.B. an Jitsy oder Bigbluebutton, so hält sich die max. Zuschauerzahl von ca. 100 in Grenzen im Vergleich zu 50k-500k Anfragen pro Sekunde, die ein moderner Webserver verarbeiten kann.

Letztendlich ergaben Recherchen, dass sich auch für WebRTC Strukturen entwickeln, die eine Skalierbarkeit und Elastizität zulassen. Das Stichwort WebRTC at Scale stimmte uns zuversichtlich, dass eine ausreichende Skalierbarkeit für die XRevent Plattform erreicht werden könnte.

Systemübersicht

Der typische Anwendungsfall für den XRevent-Broadcaster bildet sich in etwa wie folgt auf das verteilte System ab. Ein Künstler*innen erzeugt ein Videobild (meist mit Audio), welches von mehreren Konsumenten auf einer Quest, Quest2 oder Pico4 empfangen wird. Damit die Performer*innen die Reaktionen abschätzen können, erhalten sie ein Stream aus der VR oder direkt zurück.

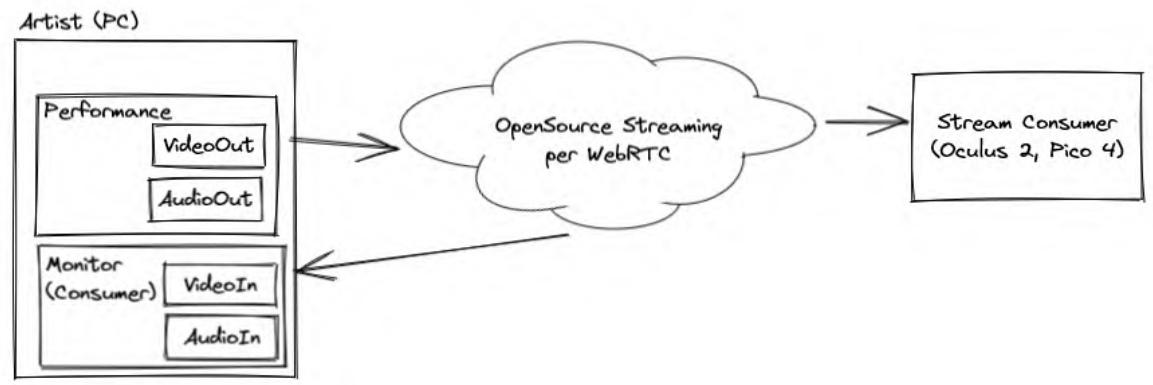


Abbildung 1Einfache Netzwerkübersicht

Um die drei Subsysteme Android Plugin, Unity App und React Web App zu detaillieren zeigt die Skizze unten hauptsächlich die Möglichkeiten an den Endpunkten.

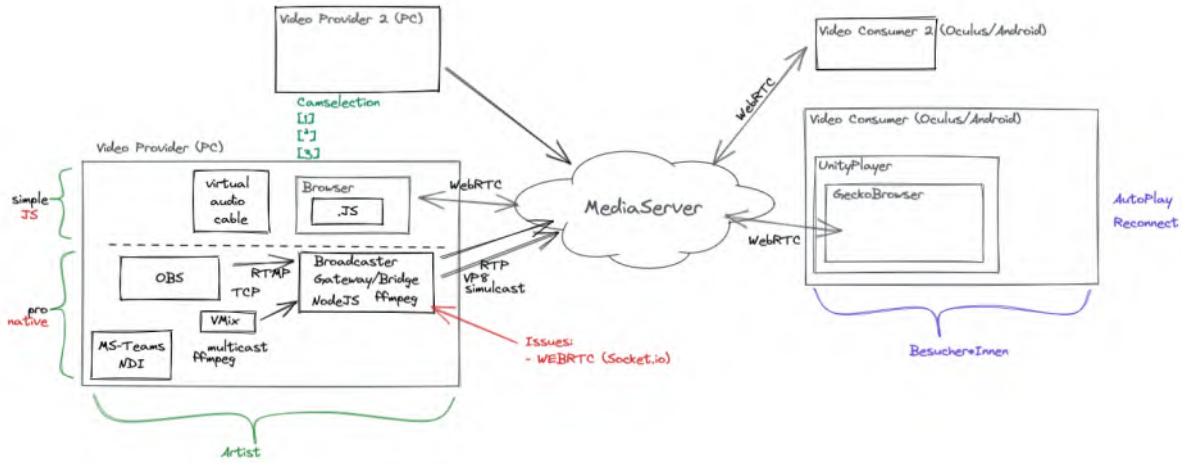


Abbildung 2Detailliertere Übersicht der Endpunkte

Zunächst wurde über eine eigene native Implementierung (In der Abbildung als „pro“ bezeichnet) des WebRTC Standards nachgedacht mittels Libraries wie z.B. Socket.io um so Kontrolle über die Entwicklung der nötigen Selective Forwarding Units (welche unter anderem für die Skalierbarkeit zuständig sind) und die plattformspezifischen Implementierungen für Unity zu haben. Diese würden sich immerhin auf Mac, Linux, Android und Windows PC erstrecken, das diese als Build Target für Unity möglich sind. Schnell wurde jedoch klar, dass so das Projekt sehr umfangreich würde.

Eine Recherche existierender Open Source Selective Forwarding Units (SFU) wurde vorgenommen, mit der Feststellung, dass auch hier sich die Projekte auf eine geringe Anzahl beschränken. Kurento, Janus, ION-SFU, MediaSoup, Galene und lib-jitsy-meet dürften hier die wesentlichen Projekte sein. Kurento z.B. ist der WebRTC Medienserver, der im wesentlichen BigBlueButton (BBB) ausmacht; MediaSoup liegt neueren Versionen von BBB zugrunde und wird auch z.B. von Mozilla Hubs verwendet. Eine Sammlung der Videokonferenz Architekturen kann unter https://github.com/xrchisense/xrevent-broadcaster-unity/wiki/Open-Source-Video-Streaming eingesehen werden.
Letztendlich fiel die Wahl auf das Open Source Projekt LiveKit. Neben dessen Performanz, war vor allem die Elastizität und Vision des Projektes ausschlaggebend. Zum Thema Elastizität lässt sich sagen, dass Livekit weitgehende Integrations-, Test- und Deploymentmöglichkeiten von WebRTC anbietet. Das Client SDK beispielsweise bedient Plattformen wie IOS, Android, Unity Web, Flutter, React und Web. Das Server SDK Java/Kotlin, Ruby, Go und Node.js. Grade die Deployment

Möglichkeiten des Servers sind aus unserer Sicht vorteilhaft für Open Source Projekte, bzw. Projekte mit wenig Budget, da dieser leicht als Development Server lokal installiert werden kann, auf einem VHost einfach selbst gehostet werden kann oder als Cloud Service gemietet um etwa einen Administrationsaufwand zu vermeiden. Beides ist also möglich: Kosten vermeiden oder Arbeitsaufwand vermeiden; gepaart mit einer breit aufgestellten Plattformvielfalt für die Clients.

In Punkto Vision geht Livekit den Gedanken der föderierten Verbreitung von Inhalten, wie man ihn etwa kennt von Systemen wie Bittorrent (Fileshare), eDonkey (Fileshare), Gnutella (Fileshare), Matrix (Chat), Mastodon (Kurznachrichten), Peertube (Video CDN) kennt.

Android Plugin

Die Quest2 von Meta ist mit 10,4 Millionen verkauften (Verkäufe von 2016 bis Q1 2021) VR Headsets weltweit Marktführer. Somit wird die virtuelle Realität hauptsächlich mittels Android Geräten und dem API Level 29 konsumiert. Betrachtet man den Technologiestack dieses ARM7 basierenden Geräts, so ergaben sich mehrere Ansatzmöglichkeiten ein Videobild rendern zu lassen.

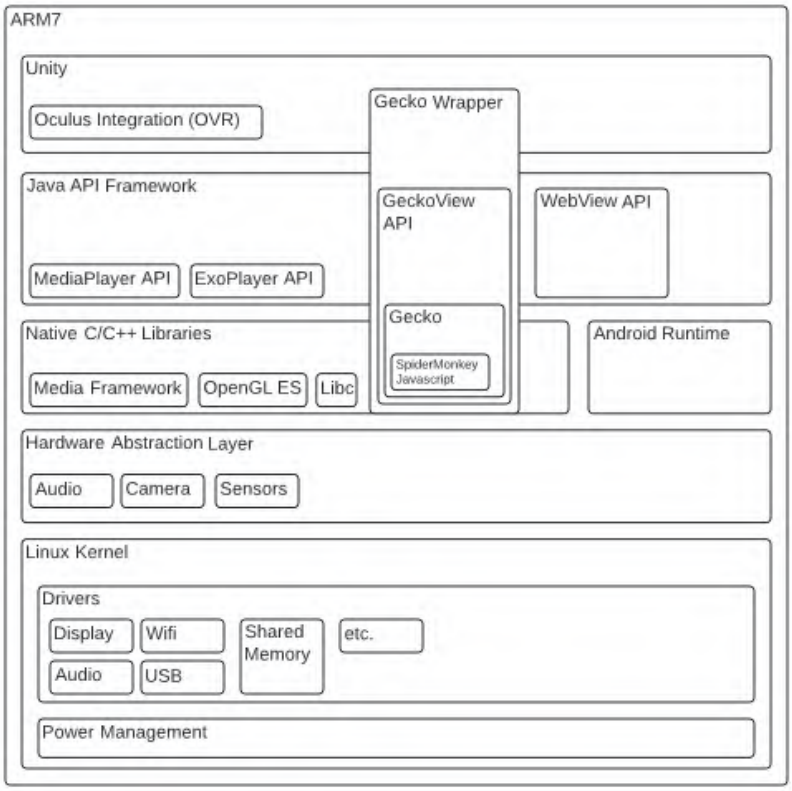


Abbildung 3Quest (Android) Technology Stack

Zum einen könnte das Android Media Framework verwendet werden. Hier würde man auf Nativer C/C++ Ebene arbeiten. Alternativ könnte man auf Java API ebene arbeiten und die MediaPlayer API, ExoPlayer API, ExoPlayer2 API oder den Android WebView nutzen. Es bieten sich also viele Möglichkeiten ein H.264 encodiertes Bild zu rendern. Wenn man sich jedoch WebRTC näher anschaut, so stellt man fest, dass es nicht „die eine“ WebRTC Implementierung gibt. WebRTC kann verschieden implementiert werden. Um sich eine Vorstellung darüber zu machen, was WebRTC eigentlich ist, hilft es sich die Architektur der WebRTC.org anzuschauen.

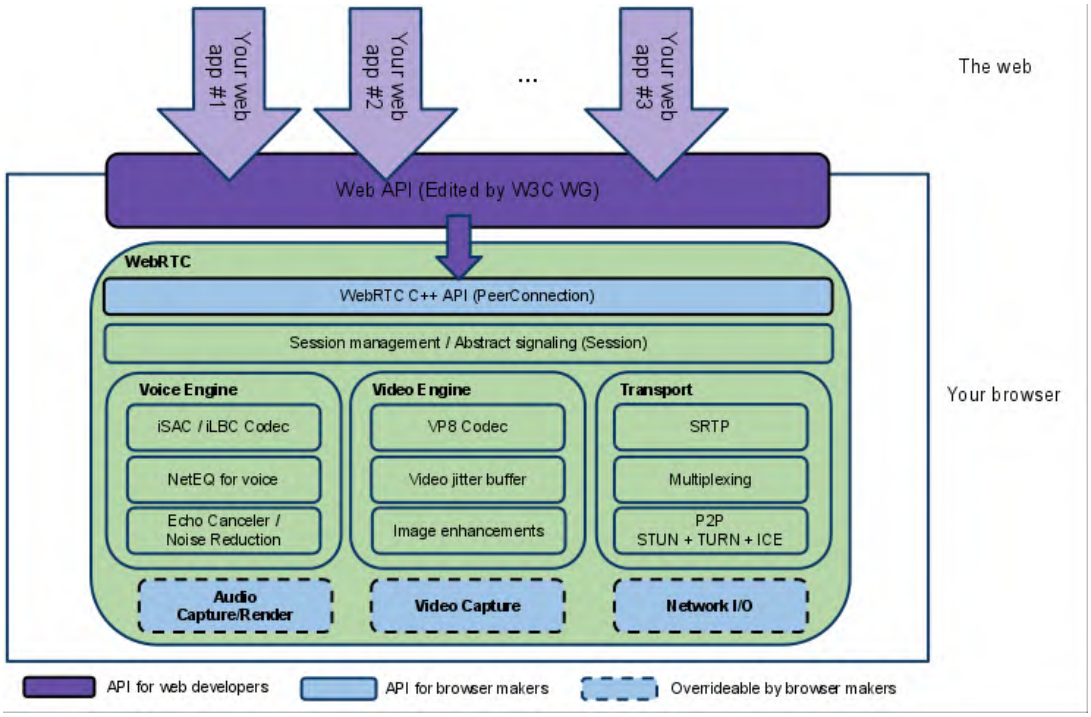


Abbildung 4WebRTC Architectur im Browser

So können Network I/O, VideoCapture, Audio Capture/Render und Peer Connection unterschiedlich implementiert werden, was wiederum Auswirkungen darauf hat, wie ein Server zu programmieren wäre. D.h. bei einem Wechsel der Cloud Infrastruktur, wäre eine neue native Implementierung notwendig. Ein Großteil den des System müsste neu programmiert werden.

Die erarbeitete Lösung hier ist, einen Browser zu nutzen anstatt einer nativen Implementierung. Mit dem GeckoView bietet Mozilla einen Android Wrapper für die Gecko Webengine an, die auch z.B. Mozillas Firefox antreibt. Im Kern des Plugins wird also ein Gecko thead erstellt mit dem Webseiten, auch wenn sie nicht sichtbar sind geladen werden können. So können auch JavaScipt basierende VideoPlayer geladen werden. Einziges Problem ist die Weitergabe des Bildes vom Gecko an Unity.

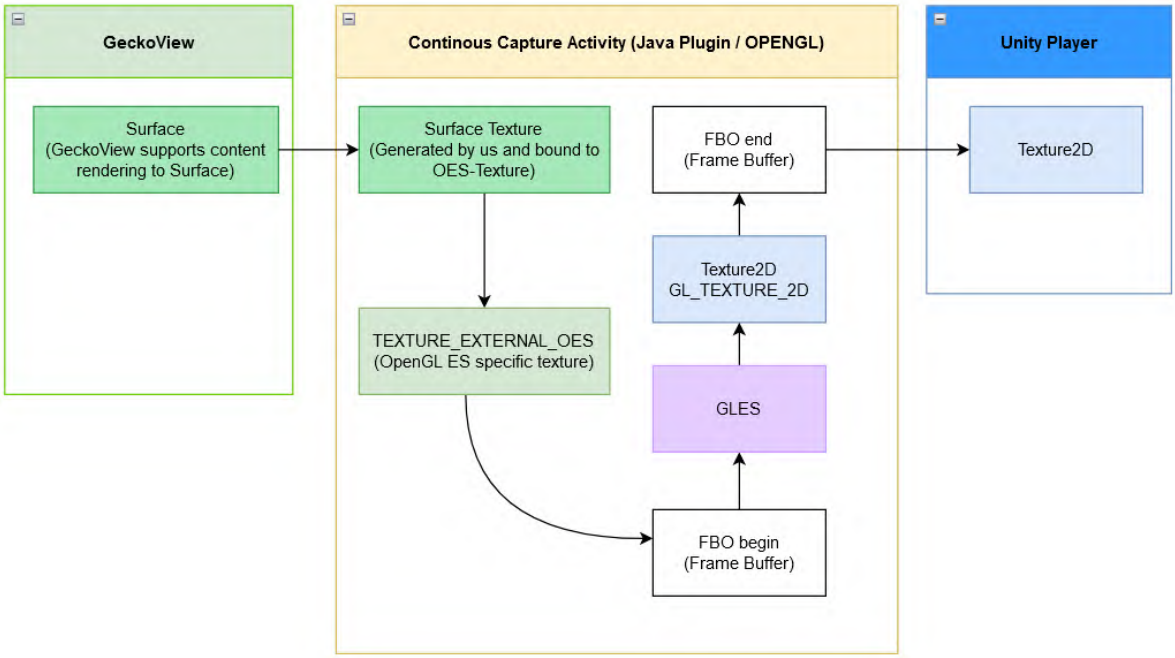


Abbildung 5Bilddaten aus dem GeckoView nach Unity per OpenGL ES Programmierung

Dies geschieht indem Android Surface Textures mittels OpenGL Framebuffer in Unity Texture2D Bilddaten konvertiert werden. Der gesamte Ablauf sieht in etwa wie folgt aus.

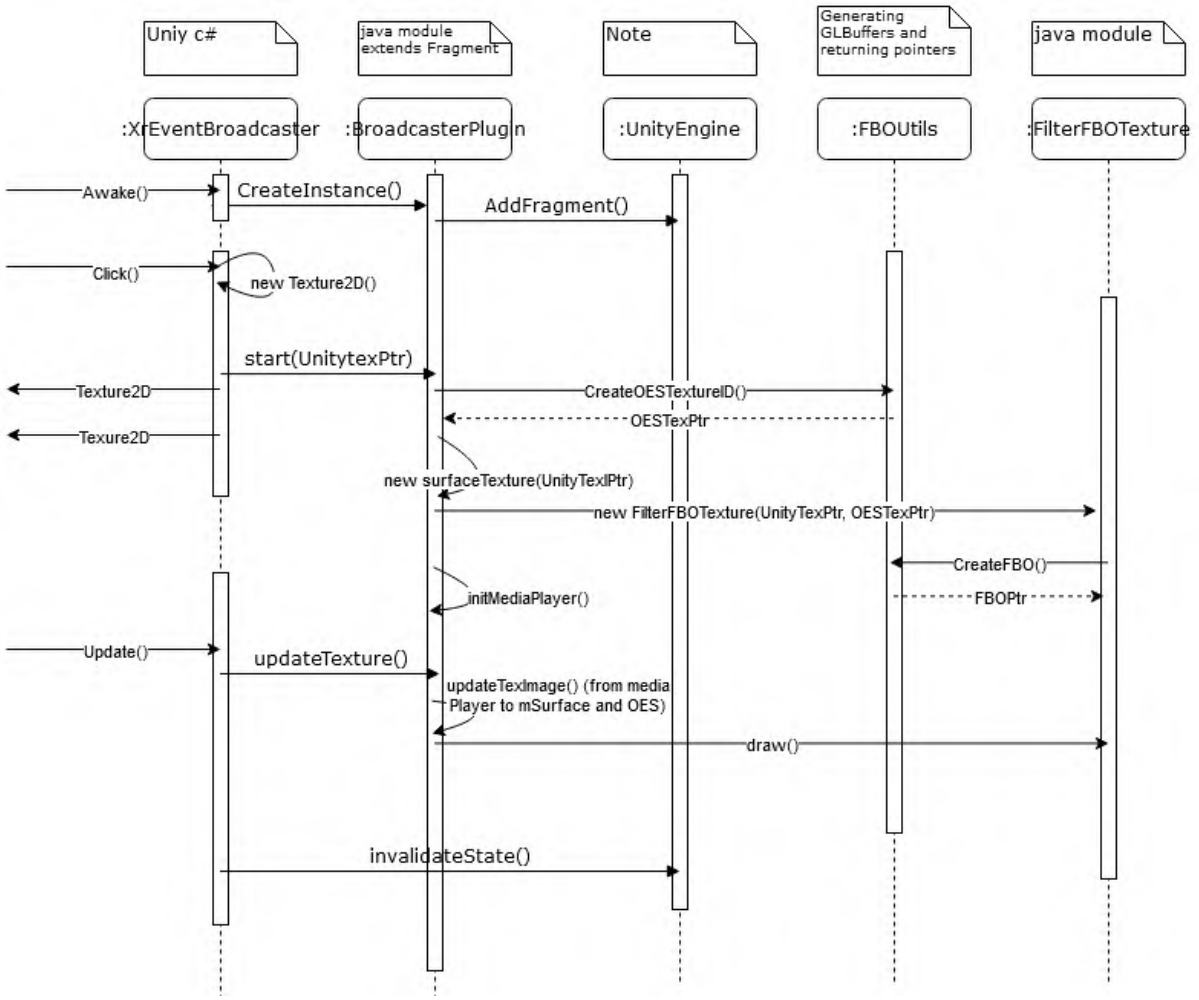


Abbildung 6 Sequenzdiagram Broadcaster

Der entsprechende Code kann unter https://github.com/xrchisense/xrevent-broadcaster-android eingesehen werden.

Unity Application

Unity bietet die Möglichkeit native Plugins (plattformspezifische Libraries) einzubinden. Dies ermöglicht es Features, wie zum Beispiel Aufrufe auf Betriebssystemebene, auszuführen, die in Unity nicht Verfügbar sind. Im Bereich Android Plugin Entwicklung, bieten sich verschiedene Möglichkeiten und es Bestand zu Anfang die Idee das vorliegende Java Plugin komplett in Unity zu entwickeln (kompilieren), um eine einfache Modifizierbarkeit und Transparenz (insbesondere für die verschiedenen Plattformen Android, Windows, Mac, etc.) des Open Source Codes zu gewährleisten. Leider wurde klar, dass die Debugmöglichkeiten sehr eingeschränkt sind und zum größten Teil ohne Debugger entwickelt werden müsste. Dies hat sich als sehr zeitraubend herausgestellt und bat auch keine Aussicht auf Erfüllung der oben genannten Ziele für etwaige Projektfortführende. Das Plugin wurde schließlich in Android Studio vorkompiliert und in Unity mittels Gradle während des Buildprozesses zusammengeführt. Laufzeitdebugging wurde in Unity auf dem Endgerät mittels LogCat durchgeführt.

Um eine möglichst einfache Anwendbarkeit für Unity Creatives zu erreichen sollte das Plugin als Unity Package bereitgestellt werden. Dies hat den Vorteil, dass User beispielsweise über den Unity Store komfortabel das Plugin laden können. Leider hält Unity für Plugins eine besondere

Ordnerstruktur vor, die im Projekt eingehalten werden muss und die Nutzung „externe Package verhindert. Da dies aber ein typisches Unity Problem ist, sind fortgeschrittene Anwender mit diesem Problem vertraut. Gerne hätten wir aber auch Anfänger besser adressiert.

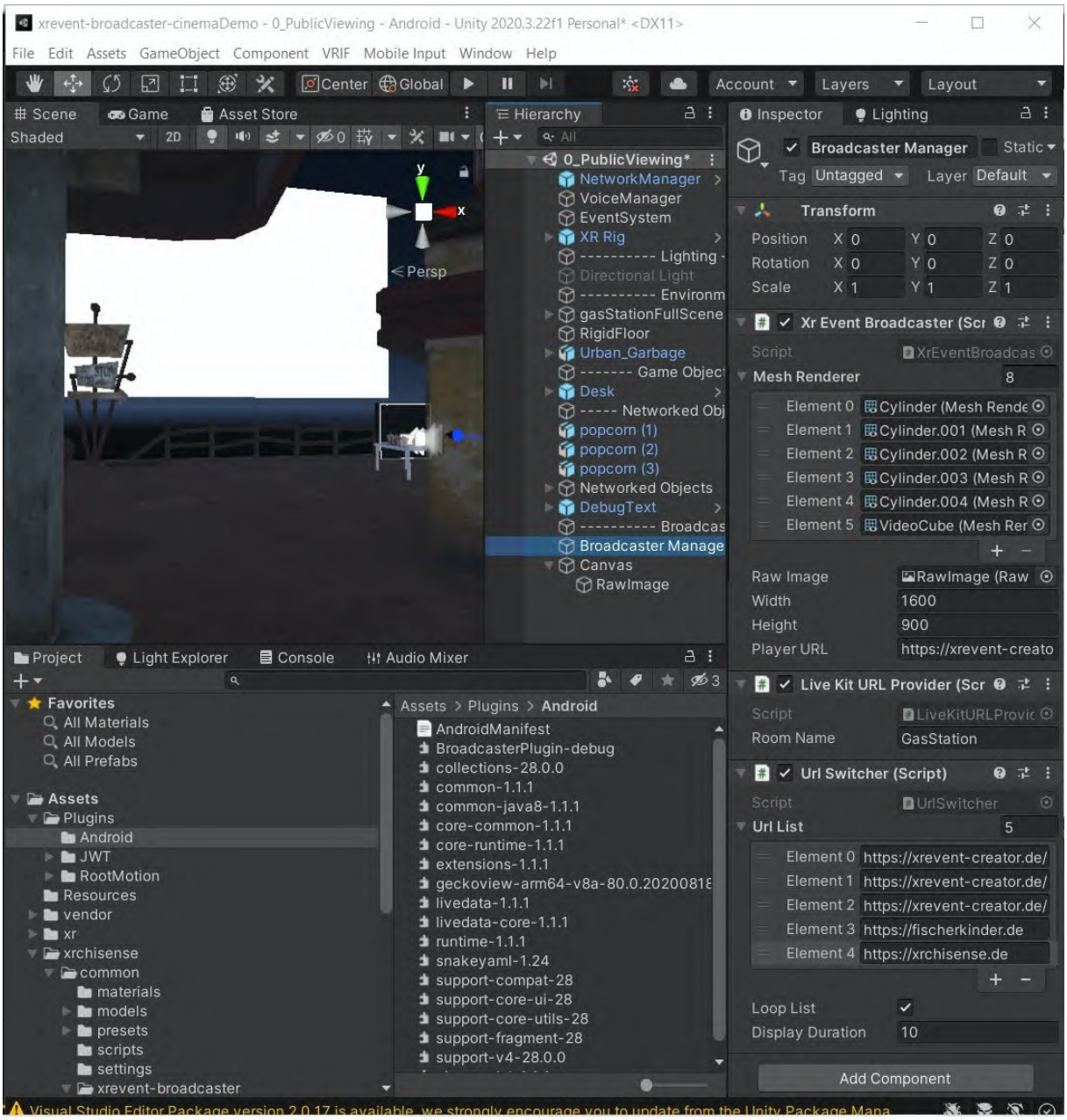


Abbildung 7Das Plugin aus Sicht des Anwenders als Komponenten

Insgesamt bietet das Plugin mehrere Möglichkeiten Video zu verwenden. Die Funktionen des Plugins werden in Form von Componenten/Skripten im Inspektor bereitgestellt (siehe Abbildung rechts). Das Hauptskript ermöglicht es GeckoView Bilddaten per Meshrenderer auf 3D Geometrien (Meshes) oder auf sog. Raw Images mittels Universal Render Pipeline (URP) und OpenGL ES 3.0 zu rendern. Dazu muss der Anwender lediglich eine Player URL in der XRevent Broadcaster Komponente angeben. Diese Player ist in unserem Prototyp ein mit dem Livekit Client SDK umgesetzte React Web App. Im Prinzip können aber jegliche URIs angegeben werden, die der GeckoView rendern kann. Das JavaScript Modul SpiderMonkey ist hier dafür verantwortlich den LiveKit JS Player zu interpretieren.

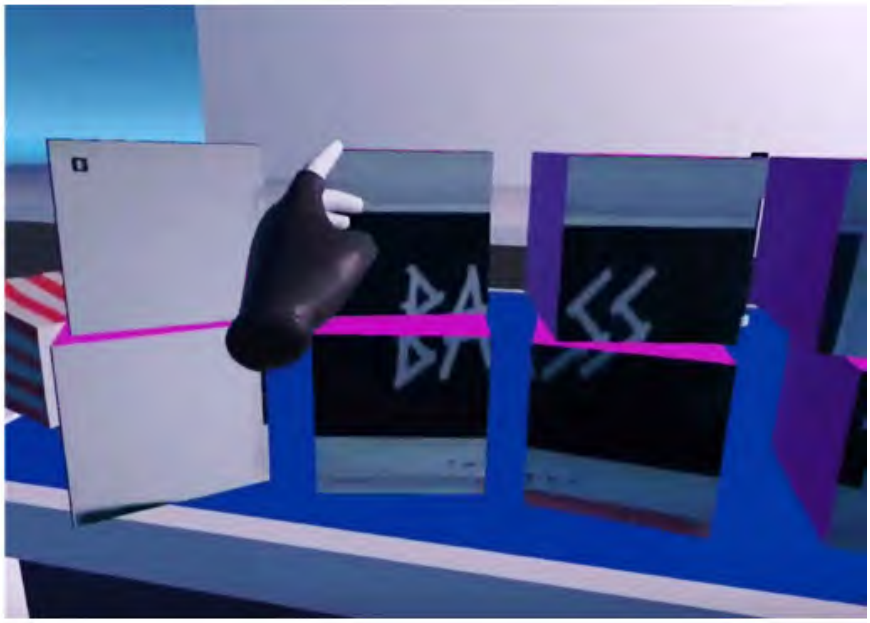


Abbildung 8Dank UV-Manipulation der Geometrien (hier Würfel) können live Bilddaten frei positioniert werden.

Zu Testzwecken der Kommunikation zwischen Unity Plugin Interface und Plugin wurde ein URL Switcher Skript geschrieben, der es erlaubt http Adressen zur Laufzeit auszutauschen. So können beispielsweise Programminformationen, Logos, Werbung, etc. dem Livestream zwischengeschaltet werden.

Der Livekit URL Provide ist für die Anmeldung am Livekit Server (Signaling Server) verantwortlich. Um eine Verbindung zu einem Raum herzustellen, in dem eine oder mehrere Videospuren publiziert werden, wird hier mithilfe eines API Keys verschiedene Metainformationen verschlüsselt, die der Server für Clientverbindungen nutzt. Dies geschieht unter Zuhilfenahme einer JWT Library die als externes Plugin auf Basis von .NET 4.x realisiert wurde. Ein Zugangstoken wird somit zur Laufzeit generiert ohne eine Verbindung zu einem Autorisierungsserver haben zu müssen.

Sollte sich die Anwender*in ebenfalls für eine Nutzung von Livekit entscheiden, so braucht er hier nur den Raumnamen anzugeben, in dem der Stream publiziert wird.

Web Application

Aufgrund der Entscheidung einen Browser als Endpunkte zu nutzen ist die eigentliche Java Script Applikation sehr flexibel änder- und austauschbar. Da die Javascripte erst zur Laufzeit in Unity Client geladen werden, besteht im Prinzip freie Kontrolle über das was auf einem RawImage oder Mesh passiert.

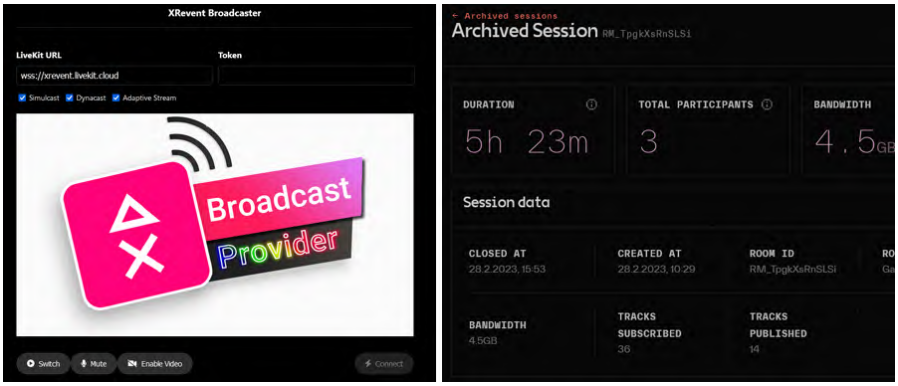


Abbildung 9Video Ingest via Browser Videoquelle (links), LiveKit Cloud Service Statistiken (rechts)

Für unsere Use Cases, welche wir in den Demos LiveSet und CineDemo verdeutlichen, nutzt ein/e Künstler*in das in React programmierte Web Frontend https://xrevent-creator.de/livekit/ im Provider Modus. Hierbei kann per OBS virtual Webcam oder einem Hardware Videograbber ein Bewegtbild direkt zur Auslieferung per WebRTC über die Livekit Infrastruktur bereitgestellt werden.

Alternativ bietet das Livekit Projekt auch die Möglichkeit Video per RTP Ingest einzuspeisen, wie man es beispielsweise von OBS Upstreams zu YouTube kennt. Das benötigte Modul Ingress-Livekit bietet somit noch einmal mehr Flexibilität beim professionellen Mixing von Bildquellen und Bild-Ton Kopplungen, denn eine virtuelle Webkamera, wie sie im Broadcast Provider genutzt wird, hat zunächst kein Audio. Dieser wurde in unserem Fall durch virtuelle Audiokabel (z.B. Voicemeeter) erzielt.

Demo Applicationen

Um konzentriert am Kern des XRevent Broadcaster arbeiten zu können und in-vivo Probleme schnell zu erkennen, wurden zwei Use Case Szenarien entwickelt. Das (1) CineDemo und (2) das LiveSet Demo. Ziel des CineDemo war es über mehrere Stunden eine unterbrechungsfreie VideoPerformance testen zu können. So wurden z.B. bei MultiUser Videoabenden getestet, inwiefern ein Realszenario mit verschiedenen Bandbreiten und Verbindungsabbrüchen für eine gemeinsame Soziale VR Group Experience taugt. Ziel des LiveSet Demos war die Demonstration der sehr geringen Latenz in einem Live VJ Set in dem mit einem Sequencer Ton und Bild Live präsentiert wird.

1	XRevent CineDemo	https://github.com/xrchisense/xrevent-broadcaster-cinemaDemo
2	liveSet Demo	https://github.com/xrchisense/xrevent-broadcaster-liveSetDemo



Abbildung 10 Multiuser CineDemo (links), GalerieSzenario mit Live.isuals aus TouchDesigner (rehts)

Dank der Möglichkeit Texturkoordinaten in Geometrien frei zu manipulieren, können neben einfachen Flächen auch nur Fragmente des Videos dargestellt werden. Somit könnte auch z.B. eine 3D VideoKonferenz leicht realisiert werden.



Abbildung 11Live Setup Beispiel: Bild und Ton können per Midi Sequencer live manipuliert werden

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Wie bereits beim vorangegangene Projekt XRevent Creator zählen vor allem Künstler*innen und Veranstalter*innen zur Hauptzielgruppe. Vor allem ihnen soll erlaubt sein Darbietungen kostenneutral umsetzen zu können und das ohne Programmiererfahrungen oder andere technischen Hürden. Dazu wird der XRevent Broadcaster in den XRevent Creator eingebunden.

Der Nutzen des XRevent Broadcasters liegt vor allem in der Reduktion von Streaminglatenzen und bietet zusätzlich eine Feedbackmöglichkeit für die Performer*innen aus dem realen Raum. Sie/er kann also im realen Raum performen und sieht fast zeitgleich wie ihre/seine Performance im virtuellen Raum beim Publikum ankommt und kann darauf reagieren. Außerdem werden so Fragerunden zwischen realer und virtueller Konferenz möglich.

In einem intensiven Coaching zur Produktenwicklung konnten wir während der Projektlaufzeit die Zielgruppe, die ein besonderes Interesse an hybriden Veranstaltungsformaten hat eingrenzen. Das sind besonders Veranstalter*innen von Konferenzen und Konzerten.

Eine Untersuchung mehrerer hybrider Konferenzen im Jahr 2020 ergab, dass der prozentuale Anteil von Frauen im Gegensatz zu den selben Konferenzen in 2018 und 2019 um 253% und von queren Teilnehmer*innen um 700% anstieg. Der von Männern hingegen nur um 121%. Ähnlich verhält es sich beim Anteil von jüngeren Teilnehmern (Anstieg von 29-42%). Das gleiche gilt für eingereichten Abstracts. So wurden um ein Viertel (26%) mehr Abstracts von weiblichen Interessierten eingereicht.[3] Die Autor*innen dieser Untersuchung führen dies vor allem auf die hohen Kosten für die Teilnahme an Konferenzen in Präsenz zurück. Die Studie kommt also zu dem Ergebnis, dass zur Förderung von Diversität und einer möglichst große Anzahl an Teilnehmer*innen den Zugang zu Wissen zu ermöglichen nur mit Hilfe von hybride Veranstaltungsformate möglich ist.

Aus eigenen Befragungen in unserem Netzwerk der Veranstaltungsbranche und Musiker*innen geben viele an, dass sie schon Erfahrungen mit virtuellen Konzerten und Festivals hatten, selbst welche durchgeführt oder Interesse haben selber welche durchzuführen. Der Grund ist vor allem eine höhere Reichweite der Darbietungen und eine gewisse Kontrolle über das eigene Werk. Außerdem interessiert viele die VR bzw. XR als neues experimentelles „Material“ für andere Darbietungsformen und zukünftige Geschäfts- oder Lizenzmodelle.

Mögliche Weiterentwicklungen, um die im Teil „Ziele“ formulierten fünf wichtigsten Aspekte für XRevent zu erreichen, sind:

Community Aufbau

Für den Aufbau einer XRevent Community wird die Webseite ergänzt, Tutorials erstellt und einen sicheren Raum für Community Treffen, Workshops und andere Veranstaltungen entwickelt. Zudem entsteht ein Manifest, welches die Werte von XRevent festhält und kommuniziert. Im nächsten Schritt werden regelmäßige Workshop Termine angeboten. Mit Hilfe von Mailinglisten und Mitteilungen zu Events und Neuerungen werden Newsletter unseres Netzwerks informiert,

[3] Skiles, M. *et al.* (2022) ‘Conference demographics and footprint changed by virtual platforms’, *Nature Sustainability*, 5(2), pp. 149–156. Available at: https://doi.org/10.1038/s41893-021-00823-2.

Multiplayer

Interessante Entwicklungsmöglichkeiten in Bezug auf Multiplayer Server und Stream Delivery sehen wir im Ansatz föderierter, dezentraler Systemstrukturen. Grade im Open Source Bereich ließen sich so Infrastrukturen von Großkonzernen im Cloud Servicebereich umgehen. VR Entwicklungen wie z.B. thirdroom.io nutzen bereits WebRTC Technologien um Textchat in VR zu ermöglichen bzw. tauschen sie Transformationsdaten über das Matrixprotokoll aus, um Positionsdaten im Multiplayer dezentral zu verarbeiten. Wie bereits beschrieben hat auch Livekit eine Vision für föderiertes live Streaming.

Profilseiten

Der Anspruch an sicherere Profilseiten für Veranstalter*innen und Besucher*innen gerecht zu werden muss ein/e erfahrene/r Entwickler*in an der Software-Architektur und der Umsetzung arbeiten. Für diese Entwicklung haben wir bereits einen Förderantrag bei NGI Assure gestellt. Die Profilseiten sollen Veranstalter*innen detailliertere Kurationsmöglichkeiten bieten. Besucher*innen erhalten u.a. die Möglichkeit Visitenkarten und Avatare zu erstellen oder Freunde und gesammelte Objekte zu verwalten.

Barrierefreiheit

Schon in den nächsten Wochen wird der XRevent Broadcaster in den XRevent Creator integriert. Außerdem soll in einem nächsten Schritt die Funktionalität des Broadcasters für PC und Mac wieder gewährleistet werden.

Hat die Arbeit in dem Projekt Dich in Deiner persönlichen, fachlichen Weiterentwicklung unterstützt?

In dieser Förderrunde waren vor allem die Coachings zum Zeitmanagement, Community Entwicklung, Open Source Geschäftsmodell und Produktentwicklung hilfreich. Sie haben das Team mit zielführenden Kompetenzen ausgestattet, um über die Projektlaufzeit hinaus Ziele für das Projekt XRevent zu formulieren und zu strukturieren.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Insbesondere die Evaluation verschiedener WebRTC Frameworks wie lib-jitsi meet, MediaSoup und Janus erforderten unsererseits eine Auseinandersetzung mit verschiedensten Softwarearchiteturansätzen und Programmiersprachen. Hier wurden schnell Testimplementierungen angestrebt um Komplexität aber vor allem Performance des Frameworks abschätzen zu können. Diese „Recherche“ Arbeit führte zwar zur Entscheidung LiveKit zu nutzen, hat aber auch jeweils viele Ressource in Bezug auf Einarbeitung benötigt.

In Bezug auf die Entwicklung der Endpunkte bzw. dem Videorendering ist ein Stand Alone Player entstanden, der Videodateien von lokalen Quellen abspielt. Die Problematik, die sich hier ergeben hat sind zunehmende Rechteeinschränkungen des Dateizugriffes auf Android, bzw. Quest2. Allerdings konnte z.B. durch die Arbeit am ThaPlayer Kenntnisse über OpenGL ES 3.0 Programmierung erworben werden.



Abbildung 12Stand Alone Unity Player (links), GeckoView im XRevent Browser (rechts)

Die Gecko Browserengine wurde zunächst als VR-Browser implementiert und Grenzen der Darstellungsmöglichkeiten (Skiptfähigkeiten) feststellen zu können. Zunächst wurde das Rendering auch über Meta's OVR Library realisiert, welche später entfernt wurde um Kompatibilität mit anderen VR Headsets, wie z.B. der Pico4 zu gewährleisten. Generell diente die Browserimplementierung dem Test der gewählten GeckoView Version.

1	ThaPlayerURPGL3	https://github.com/fischerkinder/ThaPlayerURPGL3
2	GeckoView_OVR	https://github.com/fischerkinder/GeckoView_OVR

Derzeit kann die Gecko Browserengine via GeckoView nicht in der aktuellsten Version 111.0.xx implementiert werden, da produktionsfähige LTS Versionen von Unity lediglich einen maximalen API Level von 30 in der Androidentwicklung erlauben. Experimentelle Unity Versionen bis API Level 33 existieren zwar, sind aber so nicht einsetzbar. Versuche XRevent Broadcaster mit Unity 2023.1 und aktuellstem Gecko haben ergeben, dass eine wesentliche Umstellung der Android Libraries einen wesentlichen Aufwand des Pluginrefactorings mit sich bringen werden. Aus diesem Grund wurden Entwicklungsarbeiten in diese Richtung zunächst eingestellt, da nicht produktionsfähig.

Zunächst wurde das Plugin für armeabi-v7a Architekturen entwickelt aber später auf arm64 umgestellt. Ebenso wurde der Colourspace zunächst auf linear entwickelt mit späterer Umstellung auf Gamma und OpenGL ES 2.0 wurde auf OpenGL ES 3.0 API umgestellt. Soweit steht die Zukunftsfähigkeit des Plugins relativ gut da mit dem allerdings genannten größeren Problem der Unity Android API Level Unterstützung.

Arbeitet man im Videostreamingbereich auf Low-Level Ebene, so sind FFMPEG und GStreamer die beiden wichtigsten Open Source Projekte (und leider die einzigen), mit der sich Video Encodierung und Decodierung in einer vielfältigen Art und Weise durchführen lassen. Auch hier wurden Analysearbeiten getätigt, die später keine weitere Verwendung fanden. Fest zu halten wäre, dass Open Source ffmpeg-unity-bindings Projekte existieren und auch WebRTC per Gstreamer experimentell erforscht wird.

Auch wurden eigene API Arbeiten von Unity am Unity WebRTC Modul testweise implementiert und festgestellt, dass eine hohe Perfomance in punkto Framerate vorliegt, allerdings Audio bisher vernachlässigt wurde. Starke Restriktionen führten auch hier zu einem Fallenlassen dieses Ansatzes. In Zukunft wird diese Möglichkeit aber Probleme der Plattformabhängigkeiten wohl beheben. Evtl. wird per Unity WebRTC plattformübergreifendes WebRTC (Linux, Mac, IOS, UWP, Android) möglich.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Hauptanlaufpunkt für das Projekt ist das Wiki im xrevent-broadcaster-unity repository auf github.com. Erreichbar ist dieses unter: https://github.com/xrchisense/xrevent-broadcaster-unity/wiki

Weitere Informationen zum Projekt werden auf der Projektseite https://xrchitecture.de veröffentlicht. Der XRevent Creator kann hier ausprobiert werden: https://xrevent-creator.de

Über den YouTube Kanal https://www.youtube.com/channel/UCJQsMQ1QDdgmalcNO-ldf3Q veröffentlichen wir Demo Reels, Live-Veranstaltungen und Tutorials

Über Github kann der Code eingesehen werden: https://github.com/xrchisense

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Der Arbeitsplan konnte weitestgehend eingehalten werden. Allerdings mit weitaus größeren zeitlichen Ressourcen des Teams als ursprünglich angedacht. Somit konnte der Kostenplan nicht vollständig eingehalten werden.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Fast zeitgleich mit dem Prototyp Fund Runde 12 ging im Dezember 2022 ein weiteres gefördertes Projekt durch den „Call for Transfer“ der Hamburg Innovation zu ende. Weiterhin wurden wir vom FTZ, dem Forschung und Transferzentrum der HAW Hamburg mit einem Arbeitsplatz und Infrastruktur unterstützt.

Außerdem konnten wir XRevent auf den Messen Photopia 2022, LeaTcon 2022 und Hamburg Open 2023 vorstellen. Ursprünglich war geplant zum Jahresendevent des CCC eine hybride Remoteveranstaltung in Hamburg durchzuführen. Diese konnte nach der Absage von Seiten der Veranstaltern und weniger Interesse Seitens unserer Teilnehmer an einem dezentralen Jahresendevent, der allgemeinen Energiekriese der Hochschule und damit der Abschaltung der Heizung über die Feiertage nicht mehr stattfinden.