DLR Projektträger Berlin
Eingegangen am:
09. Sep. 2021
Eingangsnr.: 5338/B0

XMPP-Media-Conf – Föderierte Audio- und Video-Konferenzen via XMPP

Schlussbericht

Zuwendungsempfänger:

Klaus Herberth

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS21S28 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Der Mensch hat ein natürliches Bedürfnis nach Kommunikation und vor allem Kommunikation in Gruppen ist ein fundamentales Verlangen. Gerade in Zeiten in denen man einander nicht sehen kann, bieten Audio- und Video-Konferenzen eine einfache Möglichkeit sich über verschiedene Themen auszutauschen. Dieses Bedürfnis befriedigen die meisten Personen über Plattformen die häufig ihren Sitz im Ausland haben und somit unklar ist wer alles Zugriff auf diese hoch sensiblen Daten hat. Gerade im Geschäftsbetrieb oder bei Behörden ist dieses Problem von höchster Bedeutung. Unser System soll genau so offen und föderal sein wie zum Beispiel die Telefonie oder E-Mail.

Um dieses Problem zu lösen wollten wir uns an bestehenden Vorschlägen in der XMPP Community orientieren und diese für die Nutzung in Gruppen entsprechend anpassen. Für die Implementierung wurde der bestehende JavaScript XMPP Client (JSXC) entsprechend erweitert. Hierzu haben wir folgende Meilensteine definiert und erfolgreich abgeschlossen.

1. Implementierung einer P2P Videokonferenz für kleine Gruppen auf Basis der bestehenden Protokollvorschläge (XEP-0272).
2. Feedback und aktuellen Stand der Implementierung des entsprechenden XEP mit der Entwickler Community teilen.
3. Evaluierung der bestehenden Videobridge mit dem Ziel eine möglichst gute A/V Qualität zu erreichen.
4. Implementierung einer Videokonferenz mit Videobridge die die Last der einzelnen Clients reduziert.
5. Ebenfalls Feedback zur weiteren Standardisierung des entsprechenden XEP-0340 an die Entwickler Community.

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

Die Zielgruppe sind alle Personen die in Gruppen via Audio/Video kommunizieren möchten oder müssen. Durch die schon bestehende breite Nutzerbasis von JSXC und die Integration in Nextcloud, Sogo, Wordpress, usw., werden die neuen Funktionen ebenfalls vielen Menschen schnell zugänglich sein.

Unser Projekt erlaubt es somit Administratoren eine offene und sichere Kommunikationslösung für ihre Nutzer*innen bereitzustellen mit einem sehr geringen Konfigurationsaufwand. Die Gewissheit, dass alle ausgetauschten Informationen innerhalb der eigenen Organisation bleiben fördert die Redefreiheit und somit eine offene Gesellschaft.

Ausführliche Darstellung der Ergebnisse

Architektur

Das Ziel einer Online-Konferenz ist es unterschiedliche Medien, wie Audio, Video, Folien, Bilder, oder Videos, mit mehreren Teilnehmenden zeitgleich zu teilen. Der einfachste Ansatz hierfür ist es, direkte Verbindungen zwischen allen Teilnehmer*innen aufzubauen (peer-to-peer, oder auch P2P abgekürzt). Dies bedeutet jede*r Teilnehmer*in sendet und empfängt einen Datenstrom an jede*n andere*n Teilnehmer*in (ein Mesh-Netzwerk). Der Vorteil dieser Methode ist, dass sie sehr einfach zu implementieren ist, wenn bereits Anrufe zwischen zwei Teilnehmenden möglich sind und keine weiteren Dienste involviert werden müssen. Ein Verbindungsaufbau ist also nur abhängig von den verwendeten Clients. Bei einer größeren Anzahl von Teilnehmer*innen stößt diese Methode aber schnell an ihre Grenzen, da ggf. der Datenstrom pro Teilnehmer*in unterschiedlich verarbeitet werden muss, was besonders bei mobilen Endgeräten oder Computern mit wenig Rechenleistung zu Problemen führen kann. Zusätzlich bieten die meisten privaten Internetanschlüsse eine asynchrone Übertragungsgeschwindigkeit, welche den Upload von Datenverbindungen, also das Senden von Medienpaketen, begrenzt. Gerade bei hochauflösenden Bildschirmübertragungen an viele Teilnehmende, kann dies zu einem limitierenden Faktor werden.

Aus diesem Grund empfiehlt es sich nach Möglichkeit, einen zentralen Vermittlungsdienst zu nutzen der die ein- und ausgehenden Medienverbindungen organisiert. Dies bedeutet, dass Clients Verbindungen zu diesem zentralen Service aufbauen und von dort die Daten von allen anderen Teilnehmer*innen erhalten. Je nach Implementierung spricht man von MCU (Multipoint Control Unit) oder SFU (Selective Forwarding Unit). MCU kombiniert beispielsweise alle Videosignale in ein einziges, indem es die Videos in einem vorgegebenen Raster anordnet und benötigt somit die geringste Bandbreite aller Möglichkeiten. Der Nachteil ist, dass das Layout der Videos vom Dienst vorgegeben wird und somit schlecht auf die jeweiligen Anforderungen des Endgeräts angepasst werden können. Zudem benötigt es für die Zusammenführung auf dem Server viel Rechenleistung. Aus diesen Gründen haben wir uns, neben der Implementierung der P2P Lösung, für die Nutzung eines SFU entschieden. Dieser Dienst verteilt alle eingehenden Signale an alle Teilnehmenden. Die*Der einzelne Teilnehmer*in sendet also nur einen Datenstrom, empfängt aber pro Teilnehmer*in einen. Dies sorgt für eine optimale Ausnutzung der asynchronen Bandbreite der meisten Internetanschlüsse. Zudem kann je nach Rechenleistung oder Verbindungsgeschwindigkeit

nur ein Teil der z. B. Videosignale angefordert werden. Dies erlaubt es auch Personen mit schlechter Internetanbindung an einer Videokonferenz teilzunehmen.

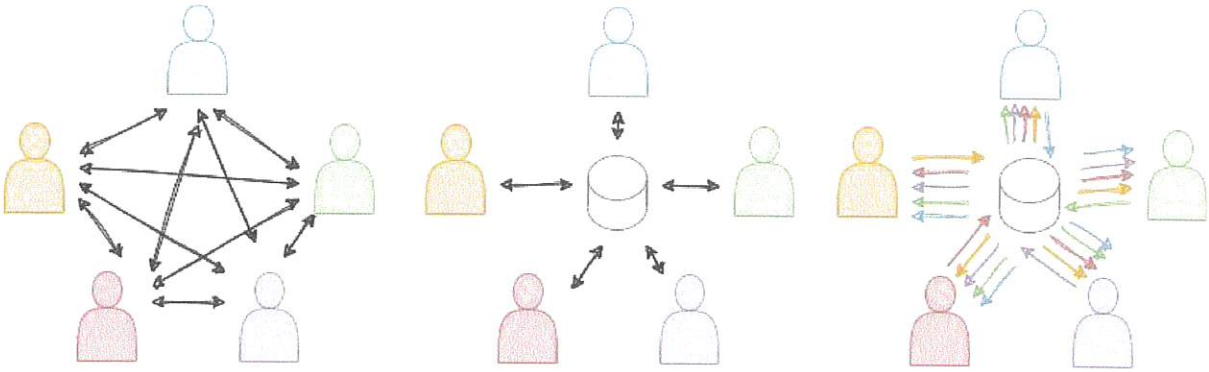


*Figure 1: Links die Darstellung eines Mesh-Netzwerks aus P2P-Verbindungen zwischen den Teilnehmenden. In der Mitte eine MCU, bei der alle Teilenehmenden exakt eine ausgehende und eingehende Verbindung haben. Bei einer SFU (rechts) muss jede*r Teilnehmende nur eine ausgehende Verbindung aufbauen, aber für jede*n Teilnehmende*n eine eingehende.*

Implementierung

Das Ziel für die Implementierung war es, den Aufwand möglichst gering zu halten und auf bestehende Techniken wie Jingle (XEP-0166) zurück zu greifen. Jingle ist bereits weit verbreitet und wird genutzt um P2P-Media-Verbindungen auszuhandeln. Um diese Technik nun für mehrere Verbindungen gleichzeitig einsetzen zu können, bestand die Herausforderung darin, allen beteiligten Konferenzteilnehmer*innen mitzuteilen, wann die Sitzung startet und die Liste aller Personen zwischen allen Mitgliedern zu synchronisieren.

Hierfür nutzen wir als Basis den Multi-User Chat (XEP-0045) mit einer nicht anonymisierten Konfiguration, da hier das Benutzermanagement schon vorhanden ist und alle Personen die gegenseitige Benutzer ID (Jabber ID) bereits kennen. Wenn in einem dieser Räume nun eine Online-Konferenz gestartet werden soll, schickt ein Mitglied eine entsprechende maschinenlesbare Nachricht (PROPOSE) in den Raum und damit an alle Mitglieder. Der Ablauf und Aufbau dieser Nachrichten ist angelehnt an Jingle Message Initiation (XEP-0353). Diejenigen, die teilnehmen möchten, senden nun eine passende "ACCEPT" Antwort. Da alle Nachrichten in derselben Reihenfolge bei allen Gruppenmitgliedern eingehen, wird nun diese genutzt um die Initiatoren der einzelnen Verbindungen festzulegen. Jedes Mitglied muss die Verbindung zu allen Personen initiieren, die ihre Antwort nach der eigenen geschickt haben. Zu beachten gilt, dass Personen nur Verbindungen von Benutzer*innen zulassen, die tatsächlich in diesem Raum sind und auch eine entsprechende Zustimmung zur Konferenz erteilt haben. Des Weiteren muss jede Nachricht eine Session ID enthalten, um mögliche Kollisionen zu unterbinden und jedes Mitglied muss seine vollständige User ID bei jeder dieser Nachrichten einfügen, da in MUC Räumen Nutzer*innen mit mehreren eigenen Geräten (multi-client support) angemeldet sein können. Zum Schließen einer Sitzung sendet jede Person eine dementsprechende Nachricht in den Raum um nachfolgenden Mitgliedern einen Überblick zu gewähren, wer gerade im Raum ist und wer nicht.

Die Implementierung für die Variante mit zentralem Dienst um Bandbreite zu sparen, funktioniert ähnlich wie der P2P-Ansatz. Es werden ähnliche Nachrichten in den Raum geschickt, aber die Nutzer*innen bauen nun nicht mit den anderen Mitgliedern eine Verbindung auf, sondern mit dem zentralen Server. Wir haben uns hier für den leicht zu erweiternden WebRTC Server Kurento entschieden der über eine Programmierschnittstelle konfiguriert werden kann. Sollte ein*e Nutzer*in sich nun dazu entscheiden, eine Konferenz zu starten, öffnet sie*er einen Kanal auf dem entsprechenden Server und startet eine Medienverbindung mit diesem. Im Anschluss veröffentlicht die Person die Kennung des Kanals im Raum. Alle Personen, die nun an der Konferenz teilnehmen möchten, öffnen selbst eine Medienverbindung mit dem Dienst und fragen alle Verbindungen von Personen, die ebenfalls teilnehmen möchten, am Server an. Im Anschluss wird in dem Raum wieder eine "ACCEPT" Nachricht gesendet, damit alle anderen Mitglieder wissen, dass sie nun die neue Verbindung anfordern können. Der Verbindungsabbau funktioniert analog zur P2P Variante.

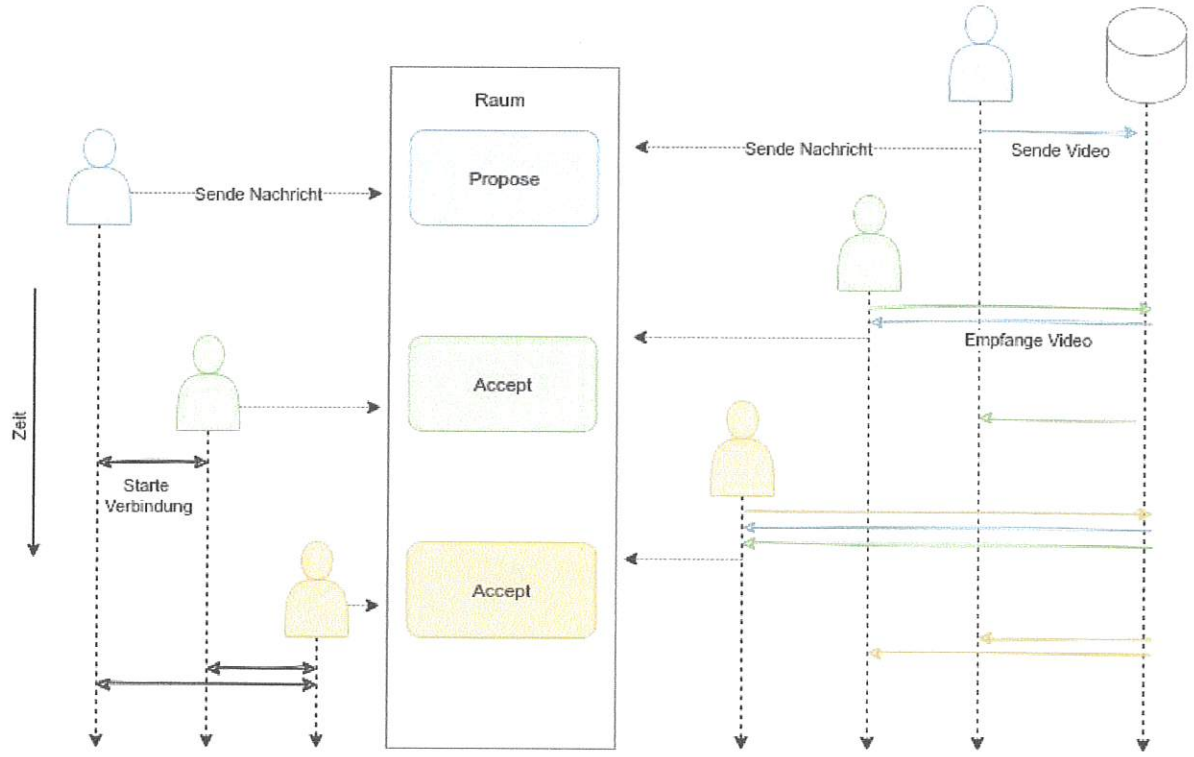


Figure 2: Die linke Seite beschreibt den Ablauf in der P2P-Variante, bei der jedes Mitglied eine Verbindung zu allen Personen aufbaut, die nach ihr der Konferenz beitreten. Die rechte Seite zeigt den Ablauf einer SFU. Wie ersichtlich ist, werden im Raum die gleichen (propose, accept) Nachrichten gesendet. Nur der Verbindungsaufbau unterscheidet sich jeweils.

Fazit

In der Projektlaufzeit konnten alle Meilensteine erreicht werden und somit können Benutzer*innen nun Audio- und Video-Konferenzen mit einem offen Protokoll nutzen. Die Begleitung durch die Open Knowledge Foundation hat gezeigt, dass eine regelmäßige Evaluierung der eigenen Arbeit und des Umsetzungsstandes elementar für eine erfolgreiche Projektumsetzung ist. Die Miteinbeziehung von externen Experten half die Probleme neu zu verstehen und andere Lösungsansätze zu entdecken.

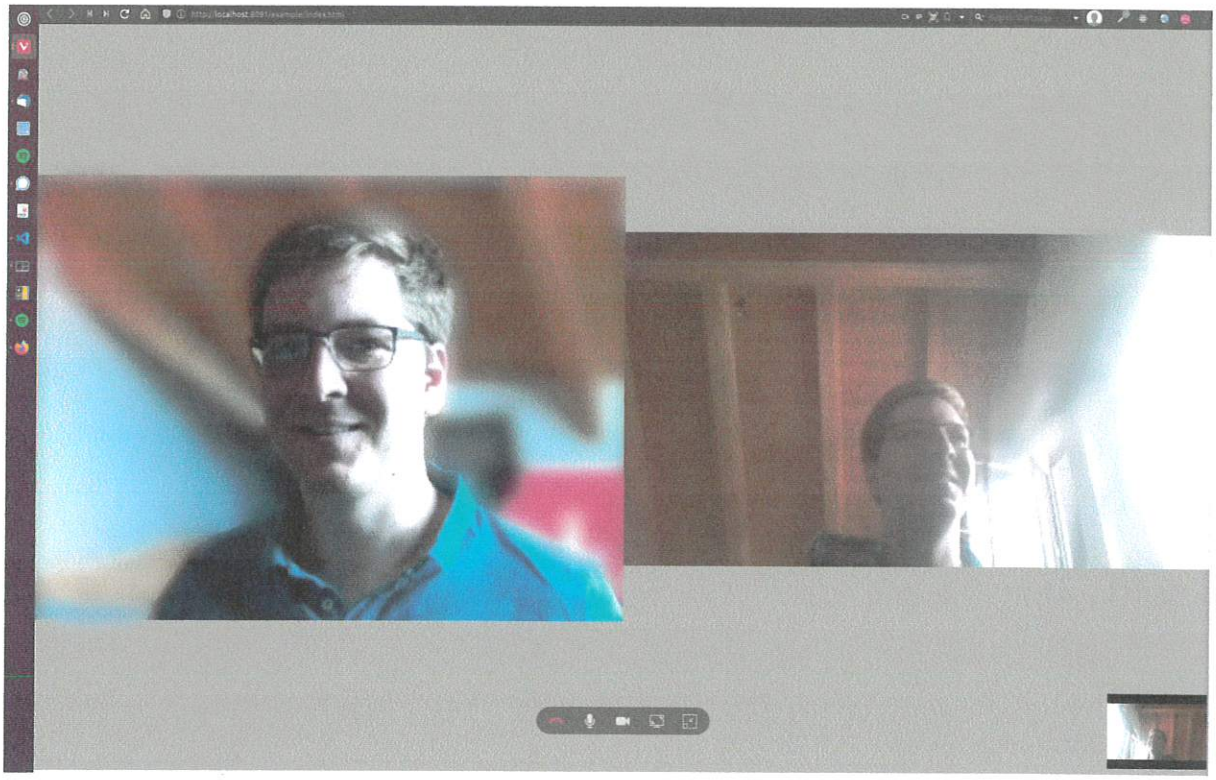


*Figure 3: Darstellung einer Videokonferenz in JSXC mit drei Teilnehmer*innen*

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Die Möglichkeit seinen eigenen Client zu nutzen um organisationsübergreifend Video-Konferenzen durchzuführen machen die tägliche Arbeit nicht nur sicherer, sondern auch einfacher und komfortabler. Um den Nutzer*innen tatsächlich eine Auswahlmöglichkeit von Clients zu bieten, sind wir zur Zeit in Absprache mit weiteren Entwickler*innen von populären XMPP Clients für sowohl Desktop, als auch Mobilgeräte. Das langfristige Ziel ist es die nun entwickelte Lösung in einen für alle offenen XMPP Standard zu überführen. Hierfür muss die aktuelle Schnittstelle zum SFU Server noch in eine XMPP Server Komponente überführt werden um einen Technologiewechsel für Client Entwickler*innen zu vermeiden.

Das Projekt hat uns ermöglicht noch tiefer in die Thematik WebRTC und Echtzeitkommunikation im Browser einzusteigen. Durch das erlangte Wissen erhoffen wir uns eine schnelle Umsetzung des Konzeptes in einen Standard.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Durch eine gute Vorbereitung konnten wir unsere Arbeitszeit zielgerichtet einsetzen und mussten keine größeren Ansätze verwerfen. Einzig die Anbindung des SFU Servers hatten wir uns ohne Zwischenschicht vorgestellt, aber in der Umsetzungsphase mussten wir feststellen, dass dies doch nötig ist und auch für eine spätere Standardisierung wichtig ist.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Interessierte können weitere Informationen im Bericht zur Demoweek des Prototype Funds[1], in unserem Blog Eintrag[2], oder im Repository[3] auf Github erhalten. Entsprechende Demos sind ebenfalls in den Artikeln verlinkt.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Auf Grund der begrenzten Projektlaufzeit mussten wir einige Komfortfunktionen wie das Hervorheben des Sprechenden, oder eine automatische Umschaltung zwischen P2P und SFU abhängig von der Gruppengröße zurückstellen. Hierdurch konnten wir im geplanten Zeit- und Kostenrahmen bleiben.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Bis auf die Diskussionen mit anderen XMPP Entwicklern, gab es keine Einflüsse von Dritten die den Verlauf des Projektes maßgeblich beeinflusst haben.

---

1 https://demoweek.prototypefund.de/projects/26-javascript-xmpp-client.html

2 https://www.jsxc.org/blog/2021/08/31/A-group-call-proposal.html

3 https://github.com/jsxc/jsxc/tree/feat-muc-media