УНИВЕРЗИТЕТ „ГОЦЕ ДЕЛЧЕВ” – ШТИП

Факултет за информатика

Александар Додевски

МОБИЛНА СОФТВЕРСКА АЛАТКА ЗА СИНХРОНА СОРАБОТКА БАЗИРАНА НА WEBRTC

магистерски труд

Штип, 2018

Комисија за оценка и одбрана

Ментор: проф. д-р Сашо Коцески
Универзитет „Гоце Делчев“ - Штип

Член: доц. д-р Васко Кокаланов
Универзитет „Гоце Делчев“ - Штип

Член: вонреден проф. д-р Александар Крстев
Универзитет „Гоце Делчев“ - Штип

Датум на одбрана: 24.12.2018 г.

Посвета или благодарност

Голема и искрена благодарност до мојот професор и ментор Сашо Коцески, кој ми пружи голема помош и разбирање во текот на целата изработка на магистерскиот труд и кој веруваше во мене и успешноста на овој труд.

Исто така, овој магистерски труд во голем дел е заради безусловната и континуирана поддршка од моите родители. Благодарност за нивната неизмерна морална поткрепа и мотивација во текот на целиот живот. Дополнително благодарност и до сите кои придонесоа во голем дел овој труд да биде успешно реализиран.

Објавени трудови

Dodevski, A., Koceska, N., Koceski, S.: A Forecasting Exchange Rate Between Macedonian Denar and Euro Using Deep Learning. Journal of Applied Economics and Business, Vol 6, No. 2, 50-61. (2018)

Мобилна софтверска алатка за синхрона соработка базирана на WebRTC

Краток извадок

Потребата за интеракција и соработка, за остварување на заеднички цели, постоела уште од почетоците на човештвото. Како што се развивала технологијата, со ист чекор следел развојот на алатките за комуникација и соработка. Брзиот развој на компјутерските системи и интернетот, основните алатки за поддршка на соработката се заменети со нови – електронски алатки за комуникација и соработка. Денес постоењето на системите во облак (cloud systems), мобилните уреди и интернетот на нештата (Internet of Things - IoT), е клучен за комплетна промена во начинот на интеракција и соработка. Денес согласно со барањата на современото општество, постојат алатки кои овозможуваат комуникација и размена на информации и содржини во реално време, придонесувајќи кон поефикасна, поефективна и поевтина соработка помеѓу тимовите кои се гео-локациски дисперзирани на планетата. Во тој контекст, новите софтверски алатки за соработка придонесуваат за големи промени во многу општествени сектори, на пример во образованието, администрацијата, индустријата, медицината, сервисните услуги и особено корисничката поддршка итн.

Денес софтверските алатки за соработка можат да се класифицираат во две категории, и тоа:

- *Асинхрони алатки за соработка*. Овие алатки им овозможуваат на учесниците да соработуваат во реализацијата на работните задачи во различни временски периоди и од различни локации.

- *Синхрони алатки за соработка*. Овие алатки им овозможуваат на учесниците да соработуваат во реално време, без разлика дали се на иста локација или дисперзирани на различни локации.

Согледувајќи ги потребите од реализација на едноставни и флексибилни синхрони алатки за соработка, главната цел на овој магистерски труд е да ги проучи постоечките алатки за синхрона соработка и да понуди архитектура на алтернативна мобилна синхрона алатка базирана на WebRTC (Web Real-Time

Communication) која ќе го минимизира користењето на системските ресурси на мобилните уреди.

За потребите на ова истражување е развиена прототип мобилна апликација која имплементира аудио и видео комуникација во реално време базирана на WebRTCи истата е искористена за проучување на најсоодветната технологија за сигнализација и пренос на податоци. Извршено е истражување преку студија на случај за квалитетот на искуство (Quality of Experience) и квалитетот на сервис (Quality of Service) на ваквиот тип на алатка, и се проучени можностите за нејзина примена како алатка за корисничка поддршка. Резултатите се елаборирани во овој магистерски запис.

Клучни зборови: *аудио и видео конференција, реално време, андроид, комуникација, корисничка поддршка*

Mobile software tool for WebRTC-based synchronous collaboration

Abstract

The need for interaction and collaboration between people with ultimate goal for achieving common goals has existed from the earliest times of humanity. As technology developed, so were the tools for communication and collaboration amongst people. The rapid development of computer systems and the Internet, the basic tools for supporting collaboration have been replaced by new electronic tools for communication and collaboration. In the present, the existence of cloud systems, mobile devices and the Internet of Things (IoT) contributes to a complete change in the way of interaction and collaboration between people. Today, in response to the needs of modern society, there are tools that enable communication and exchange of information and content in real time, and thus contribute to more efficient, effective and cheaper collaboration between people and teams geolocation dispersed on different sides of the world. In this context, the new software collaboration tools have contributed to major changes in many social sectors such as education, administration, industry, medicine, services, in particular customer support, and so on.

Today, software collaboration tools can be classified into two categories:

• Asynchronous collaboration tools. These tools allow participants to collaborate in the execution of work tasks at different time periods and from different locations.

• Synchronous collaboration tools. These tools allow participants to collaborate in real time, whether on the same site or dispersed in different locations.

Recognizing the needs of implementing simple and flexible synchronous collaboration tools, the main goal of this Master thesis is to study existing synchronous collaboration tools, and to offer the architecture of an alternate mobile synchronous tool based on WebRTC (Web Real-Time Communication) which will minimize the use of system resources on mobile devices and provide better performance while using multiple users.

For the needs of this research, a prototype mobile application has been developed which implements real-time audio and video communication based on WebRTC and

it is used to study the most appropriate signaling and data transmission technology. Research was carried out through the Quality of Experience case study and the quality of service of this type of tool, and studied the possibilities for its application as a tool for custmer support. The results are elaborated in this Master thesis.

Key Words: Audio and video conference, real time, android, communication, customer support

Содржина

1. Вовед

Динамичноста на светот во кој живееме побарува константен развој и надоградба на начините за соработка. Постојано се работи на подобрување на старите и откривање на нови технологии за поврзување, комуникација и колаборација.

Остварувањето на најразлични цели, индивидуални или општествени ја истакнува потребата и наклонетоста кон воспоставување на комуникација и реализирање на соработка. Некои од нас преферираат електронска пошта за комуникација, други телефонски или видеоконференциски повик, некои пак лице в лице состанок, или пак комбинација од повеќе. Генерално, може да сепарираме два начина на комуникација: синхрона - комуникација во ист временски период т.е. во реално време и асинхрона – комуникација во различен временски период. За двата начина локацијата не е важна.

1.1. Синхрона комуникација

Етимолошки гледано зборот synchronous е конкатенација од два збора syn + chrono, што означува заедништво во време. Тоа значи дека два или повеќе комуникациски системи се координираат да бидат присутни (не мора локациски на исто место) во ист временски период за да соработуваат.

Историски гледано, синхроната комуникација на почетокот била достапна само со кажан збор или знак користејќи сигнали. Телеграфот, а подоцна телефонот ја проширил синхроната комуникација. Радио комуникацијата почнала да го отстранува ограничувањето во однос на локацијата, дозволувајќи точките за комуникација да се поставени секаде, доколку ја имаат соодветната опрема за испраќање и примање на сигнали. Во модерната дигитална ера синхроната комуникација вклучува сателити, мобилни телефони и интернет технологии и овозможува соработка во реално време без оглед на локацијата. Најнов тренд во областа на синхрона комуникација во последните неколку години е WebRTC технологија, т.е. веб комуникација во реално време. Facebook Messenger како една од моментално најактуелни апликации како мотор користи WEBRTC во позадина.

Синхроната комуникација се интерпретира преку: телефонски повик, видео и аудио повик, конференциски повик, чет и слично.

Придобивките од користењето на синхроната комуникација е тешко да се набројат, еве неколку клучни:

- можноста за соработка во реално време,
- инстантен брз одговор и feedback,
- бесплатни и поевтини решенија,
- видеоконференцијата дозволува да се препознае јазик на телото (body language) и тонови на гласот и е најдобро решение за комуникација од тип еден на еден.

Како и секоја технологија истата има неколку негативности:

- Интернет пристапот мора да е стабилен и континуиран (поврзувањето е невозможно без интернет),
- Комуникација на повеќе точки во исто време бара повеќе конекции, а со тоа истите да се поврзани со минимум критериуми.

1.2. Асинхрона комуникација

Асинхроната комуникација за прв пат била развиена кога се вовел пишан збор. Асинхроната комуникација не се одвива во ист временски период и не е потребно присуство на иста локација.

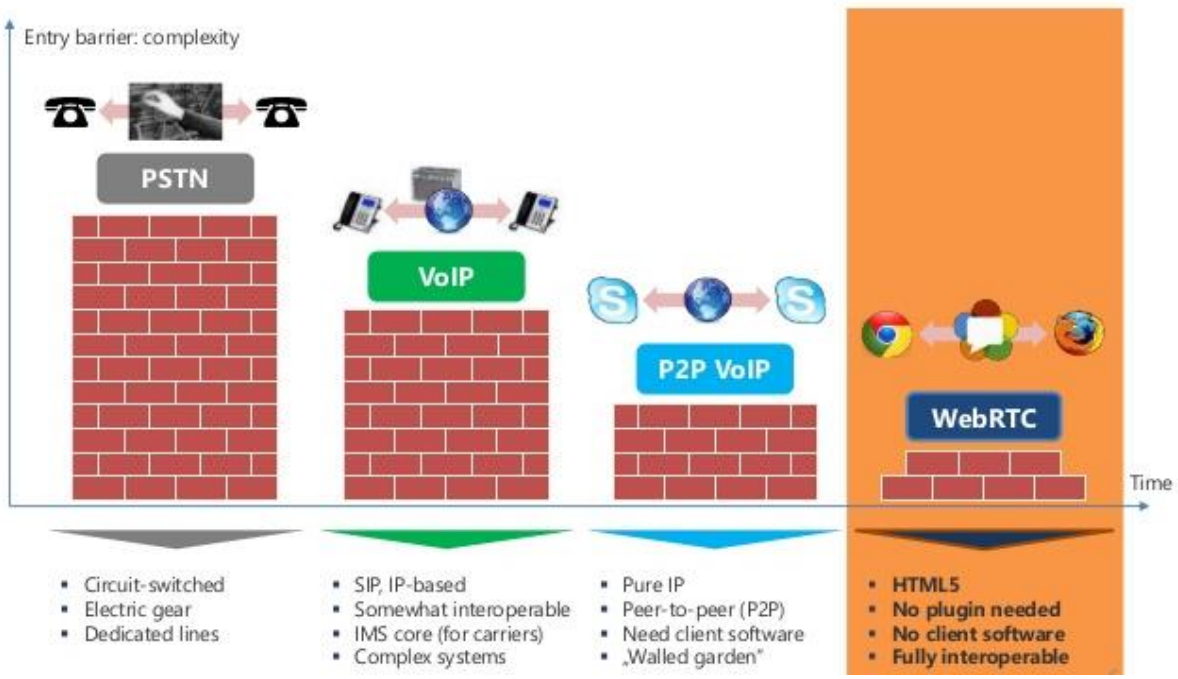
Асинхрона комуникација се интерпретира преку разни форми и тоа: електронска пошта, писмо, порака, блог, видео и аудио запис и слично.

Изборот на типот на комуникација зависи од желбата, но и од околностите и потребите на ситуацијата и можностите во моментот.

2. WEBRTC технологија

Слика 1. Лого на WebRTC
Figure 1. WebRTC logo

Не така одамна комуникацијата во реално време се сведувала на телефонски повик. Денес во областа на real-time комуникација најмногу се користи WebRTC технологијата.



Слика 2. Комуникација во реално време низ историјата

Figure 2. Real-time communication throughout history

Кратенка од WebRTC е web real-time communication при што буквалниот превод сугерира на комуникација во реално време базирана на веб. Истата

спаѓа во групата на синхрони алатки за комуникација. WebRTC овозможува директна комуникација помеѓу пребарувачите на десктоп и мобилни уреди, како и помеѓу мобилните апликации.

WebRTC е технологија популарна за аудио и видео комуникација преку пребарувачи и преку мобилни апликации, без дополнителни инсталации на интерни и екстерни плагини и додатоци. WebRTC претставува бесплатен и отворен проект, кој овозможува веб пребарувачите и мобилните апликации да комуницираат во реално време преку едноставен интерфејс, API (Application programming interface).

Слика 3. WebRTC помеѓу различни типови на уреди

Figure 3. WebRTC between different types of devices

Креирани се многу WeRTC апликации за веб пребарувачи, мобилни платформи и IoT (Internet of things) уреди, кои овозможуваат комуникација преку заеднички сет од протоколи.

2.1. Почетоци/историја

Во мај 2010 година Google ја има аквизирано корпорацијата Global IP Solutions (GIPS), која се занимава со аудио и видео пренос во реално време. Подоцна, Google има развиено многу компоненти кои се неопходни за комуникација во реално време. На пример, ги има развиено кодеците и исфрлувачите на ехо во аудио сигнал. Понатаму, Google го отвори кодот кој беше развиен од GIPS и го поврза со стандардите од W3C и IETF. Веќе во мај

2011 г. од страна на Ericsson (компанија за мрежа и телекомуникација) е креирана и претставена првата имплементација на WEBRTC.

Во октомври 2011 година W3C ја објави првата драфт верзија за спецификацијата за WebRTC.

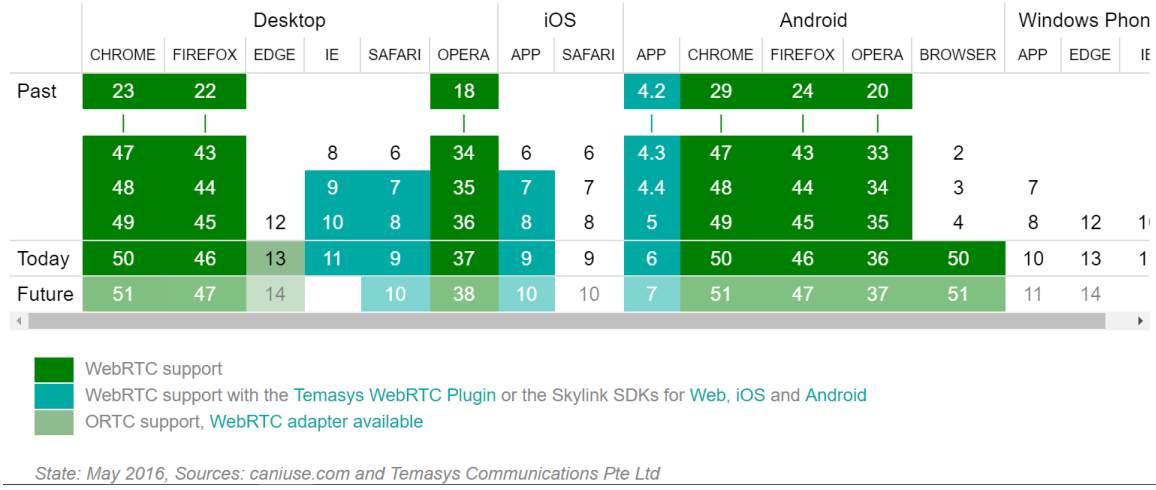
Првите чекори на WebRTC го вклучува првиот видеоповик од пребарувач (февруари 2013 г.), првиот пренос на податоци од пребарувач (февруари 2014 г.), за да во јули 2014 г. Google го претстави Hangouts како вид на апликација која користи WebRTC технологија.

Веќе во 2013 година пребарувачите за Андроид Firefox, Chrome и Opera овозможиле поддршка за WebRTC.

2.2. Компатибилност (поддршка)

WebRTC е компатибилен во повеќето од пребарувачите и мобилни околини. На следниот линк може да се провери компатибилноста на кој било пребарувач: http://www.netscan.co/demo/.

На сликата се прикажани компатибилностите за WebRTC:



Слика 4. Компатибилност на WebRTC
Figure 4. WebRTC Compatibility

Имено, поддршката на WebRTC на мобилни уреди е голем напредок, бидејќи мобилните уреди имаат ограничен екран со ограничени ресурси.

- **Android**

Во 2013 г. е претставен Firefox веб пребарувач и за андроид кој поддржува WebRTC. Така што се овозможи да се реализира видеоповик на андроид уреди користејќи Firefox мобилен пребарувач. Google Chrome и Opera за Андроид исто така подржуваат WebRTC.

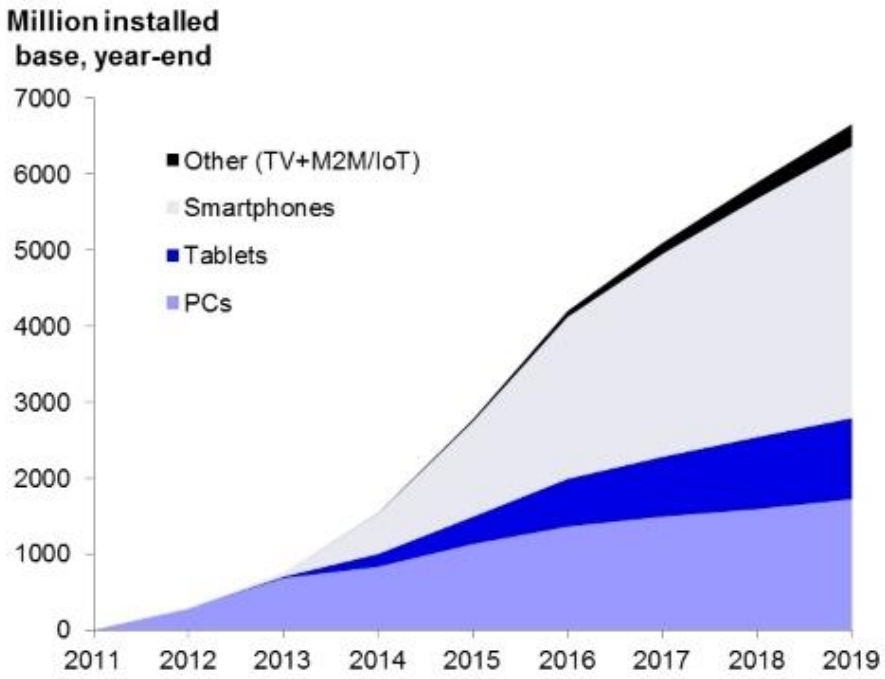
- **IOS**

WebRTC не е поддржан на IOS за пребарувачи сега за сега. Единствена опција е преку пребарувач развиен од Ericsson или пак да се креира native апликација.

- **Windows Phones**

Microsoft не поддржува WebRTC на мобилни платформи. Корисниците на windows мобилни уреди не можат да користат WeRTC апликации. Секако, поддршката за оперативниот систем Windows Phone е прекината и од Microsoft.

Сликата е прикажан графикон со растот на поддршка за WebRTC од 2011 г. до денес.



Слика 5. Графикон на користеност на WEBRTCкај различни уреди

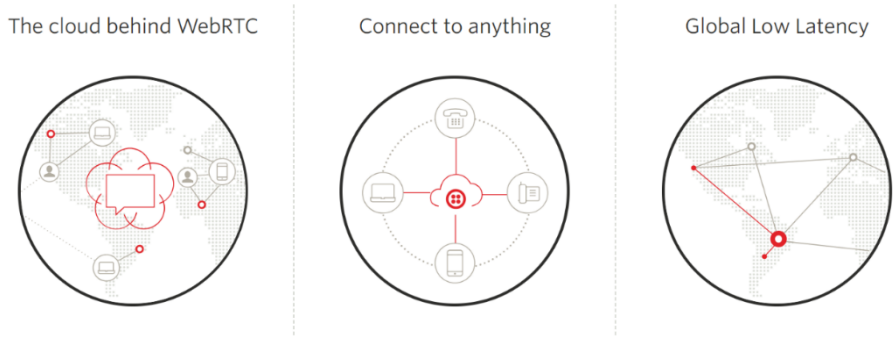
Figure 5. Graph of WEBRTC use on different devices

2.3. Карактеристики на WebRTC

Во прилог се претставени основни карактеристики на WebRTC технологијата:

1. Open source – Може секој да пристапи до кодот и да го користи за приватни или комерцијални цели.
2. Бесплатен код – Со што се намалуваат трошоците за дополнителни плагини и алатки.
3. Лесна и едноставна употреба, без потреба од дополнителни инсталации на плагини и екстензии. Не се потребни ниту Flash, Java, Silverlight слично.
4. Аудио и видео квалитет - WEBRTC користи Opus аудио кодек и VP8 видео кодек кои обезбедуваат висококвалитетни аудио и видео сигнали.
5. Безбедна комуникација - Технологијата вклучува трансфер на приватни осетливи информации, со тоа мерките за безбедност се големи за заштитата на корисниците од компромитација на податоци или изложеност на напади. Благодарение на end-to-end енкрипција, комуникацијата е приближно до 100 % безбедна. WEBRTC апликациите не треба да инсталираат дополнителни плагини за да обезбедат сигурна мрежна комуникација. WEBRTC компонените користат стандарди (RTC безбедносен протокол) за енкрипција и декрипција кои обезбедуваат заштита за време на трансмисии. WEBRTC, исто така, обезбедува DTLS и SRTP енкрипциски методи, намалувајќи го ризикот од прислушување. Дополнително девелоперите можат да ја подобрат безбедноста со користење на уште побезбедни протоколи и механизам за сигнализација. Сите конекции се заштитени HTTPS иенкриптирани (SRTP).
6. Ниска латентност при дистрибуција на податоци. Постои многу мало доцнење на аудио и видео сигналот и со тоа квалитетен видеоповик.
7. Не се потребни софистицирани телекомуникациски знаења и опрема за да употреби WEBRTC.
8. Мултиплатформа (независност од платформа и уреди) – Адаптиран за компјутери, телефони, таблети, IoT. Корисниците можат да комуницираат без да се зависни од оперативниот систем или уредот. Ова е овозможно со имплементацијата на IETF протоколи и W3C API.

9. Развивачите не се повеќе поврзани за конкретни стандарди, лиценца и vendor-specificсофтвер.
10. Универзална поврзаност – WebRTC нема ограничување во однос на типот на уредот, на пример мобилен телефон може да комуницира со таблет. Со тоа овозможува универзална поврзаност на сите уреди.
11. Облак технологија во позадина.
12. Вградена поддршка за намалување на шумот и ехото.



Слика 6. Карактеристики на WebRTC: универзална поврзаност, облак технологија, ниска латентност
Figure 6. Features of WebRTC: universal connectivity, cloud technology, low latency

Освен предности, WebRTC има и недостатоци:

1. Не е многу добар избор за еден на многу комуникација (1:M) или повеќе на повеќе комуникација (M:M), туку многу подобра опција е за еден на еден (1:1 т.е P2P);
2. Групните конференции бараат повеќе ресурси;
3. Desktop sharing е достапно и дозволува споделување на екранот со другиот корисник, но не дозволува интеракција од страна на корисникот со кој се споделува екранот, туку само следење;
4. Browser limitation – не е поддржан во Safari и некои од мобилните пребарувачи кои се постари.

2.4. WebRTC API

WebRTCAPI (Application Protocol Interface) се состои од три главни компоненти:

- RTCgetUserMedia (или само getUserMedia) – Оваа компонента овозможува пребарувачот или мобилната апликација да има пристап до камерата и микрофонот;
- RTCpeerConnection (или само peerConnection) – Оваа компонента овозможува директна комуникација на крајните корисници. Таа овозможува да се реализираат аудио и видео повици помеѓу пребарувачите или мобилните апликации. Оваа компонента ја сетира конекцијата.
- RtcDataChanels (или само dataChanels) – Оваа компонента овозможува пренос на податоци помеѓу пребарувачите или мобилните апликации, преку peer-to-peer. Сите податоци се криптирани со DTLS-SRTP

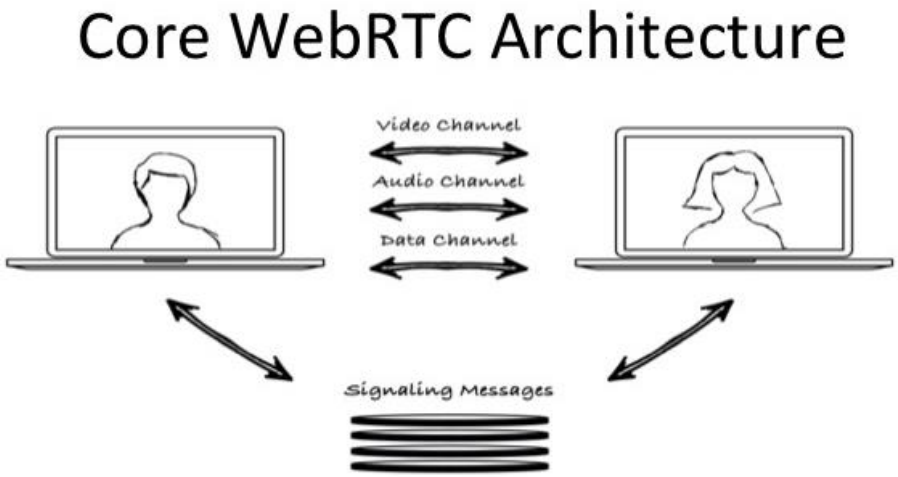
Слика 7. WebRTC API
Figure 7. WebRTC API

WebRTC ја користи RTCPeerConnection компонентата за комуникација меѓу пребарувачите и апликациите (peers), но освен тоа потребен е некој механизам (сервер) кој ќе го координира целиот тек на комуникацијата и ќе праќа одредени контролни пораки. Тој процес е познат како сигнализирање (signaling).

Методите на сигнализирање не се дефенирани од WebRTC, односно овој процес не е дел од RTCPeerConnection компонентата. Развивачите на апликации имаат можност самите тие да изберат протокол за комуникација со контролните пораки кој ќе го користат.

Постојат три типа на информации коишто може да се праќаат при овој процес и тоа: контролни пораки за одржување на сесијата (пораки за

иницијализација или затворање на комуникација, како и извештај на грешки), мрежна комуникација (пораки кои ја прикажуваат IP адресата и портот – ICE протокол) и мултимедијални можности (пораки за тоа кои кодеци и резолуции можат да бидат користени во пребарувачот – SDP протокол).

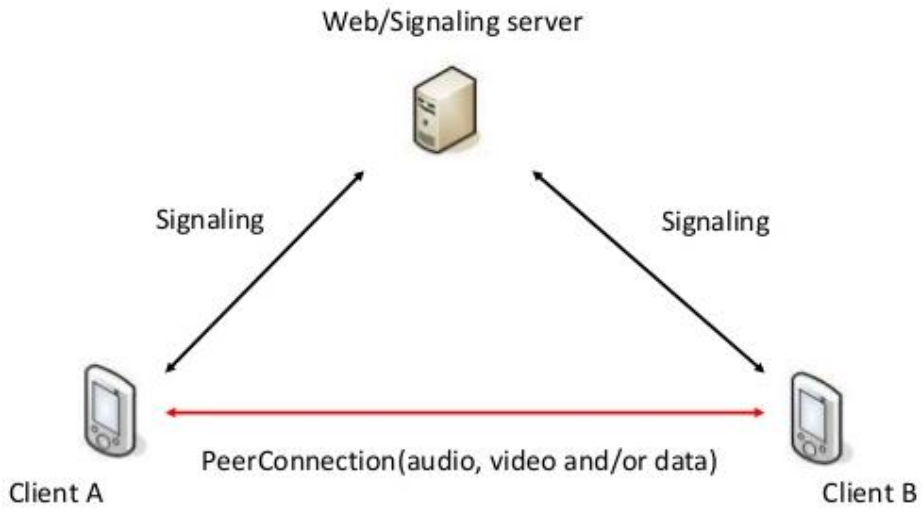


Слика 8. WebRTC архитектура за сигнализација
Figure 8. WebRTC architecture for signaling

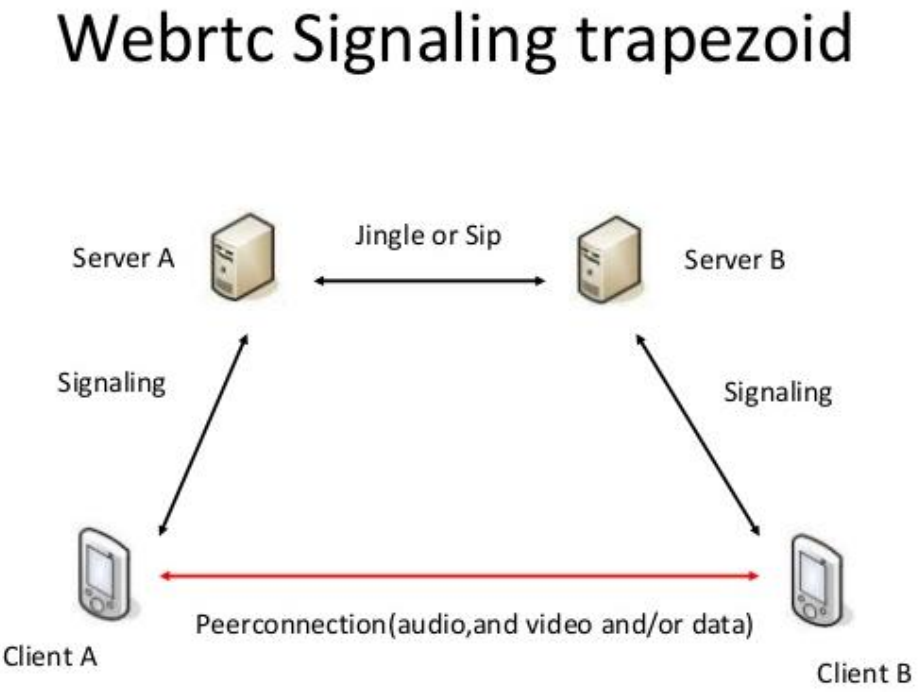
Размената на податоци со помош на сигнализирањето мора претходно да заврши успешно пред да се оствари врската.

Најчесто користен протокол за сигнализација е SIP (Session Initiation Protocol) преку Websockets или JavaScript Object Notation (JSON) преку XMLHttpRe-quest (XHR).

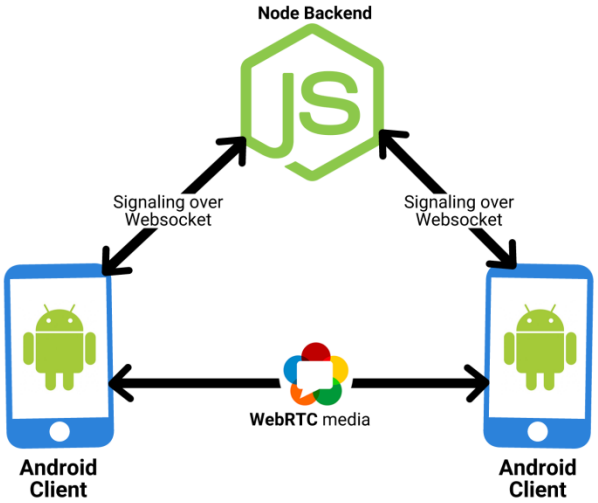
При сигнализирање доколку се користи еден ист сервер за преземање на апликацијата станува збор за триаголник сигнализација, а ако двата уреда апликацијата ја преземаат од различни сервери станува збор за трапезоид сигнализација.



Слика 9. WebRTC триаголник сигнализација
Figure 9. WebRTC triangle signaling



Слика 10. WebRTC трапезоид сигнализација
Figure 10. WebRTC trapezoid signaling

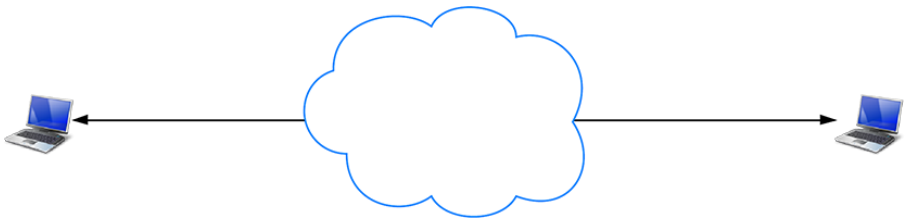


Слика 11. WebRTC сигнализација со Websocket
Figure 11. WebRTC signaling using Websocket

2.5. ICE

ICE е протокол кој се користи во процесот на сигнализација при WEBRTC, во делот кога треба да се обезбедат соодветни информации за конекциски крајни точки (peers), пред истите да реализираат директна (p2p) размена на податоци. ICE овозможува на крајните точки (peers) да се обезбеди доволно информации за топологијата на мрежата и да го најде најдобриот пат за комуникација. Користењето на ICE претставува еден вид на безбедносна мерка, бидејќи превенира да се испратат податоци на страници и апликации кои не се подготвени да ги примаат податоците.

Доколку светот би бил без NAT и firewall, при WEBRTC комуникацијата крајните точки ќе користат уникатна адреса која ќе ја разменуваат за да комуницираат директно.



Слика 12. Свет без NATs и firewall
Figure 12. World without NATs and firewall

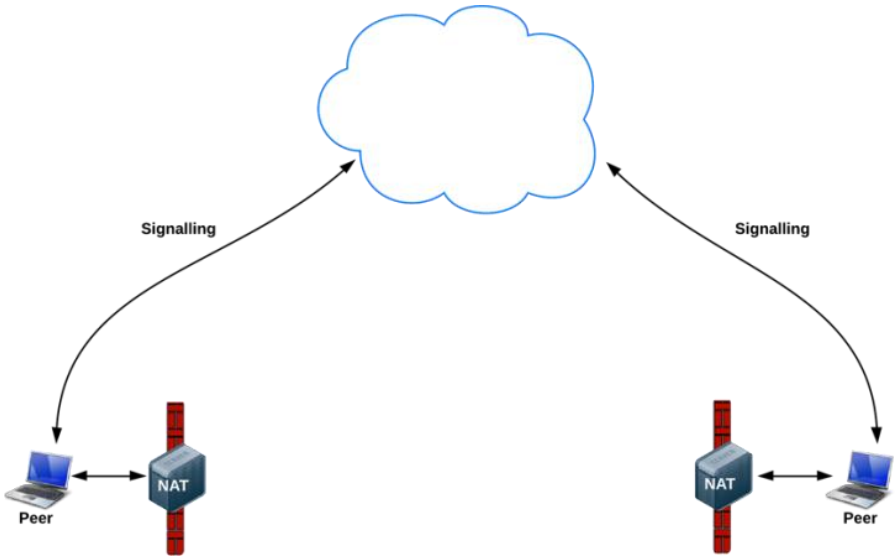
Сепак во реалноста повеќето уреди имаат повеќе слоеви на NAT, имаат антивирусни програми кои ги блокираат одредени порти и протоколи или пак користат прокси и имаат корпоративни firewall.

WebRTC со помош на ICE проколот се справува со комплексните мрежи во реалноста. ICE го пронаоѓа најдобриот пат за конекција. Пробува паралелно повеќе опции и ја избира најдобрата. Процесот за пронаоѓање на најдобар пат се нарекува NATtraversal.

ICE најпрво пробува да овозможи конекција користејќи ги хостадресите на оперативниот систем на уредите и мрежната картичка. Ако оваа конекција не успее, ICE се обидува со надворешни јавни адреси користејќи STUN сервер. Ако и оваа конекција не успее, сообраќајот се рутира преку TURN сервер.

Секој TURN сервер поддржува STUN. TURN сервер е STUN сервер со додадена способност за препраќање.

URLза STUN и/или TURN серверите се (опционално) специфицирани од WebRTC апликацијата во *iceServers* configurationobject кој е прв аргумент на RTCPeerConnection конструкторот.



Слика 13. Реален свет со NAT
Figure 13. Real world with NATs

2.5.1. STUN - Session Traversal Utilities for NAT (STUN) protocol (RFC5389)

NAT овозможува уред со IP адреса за употреба во приватна локална мрежа, но оваа адреса не може да се користи надвор од таа мрежа. Без јавна адреса крајните точки при WEBRTC комуникација не би можеле да комуницираат. За да се надмине тоа WEBRTC користи STUN.

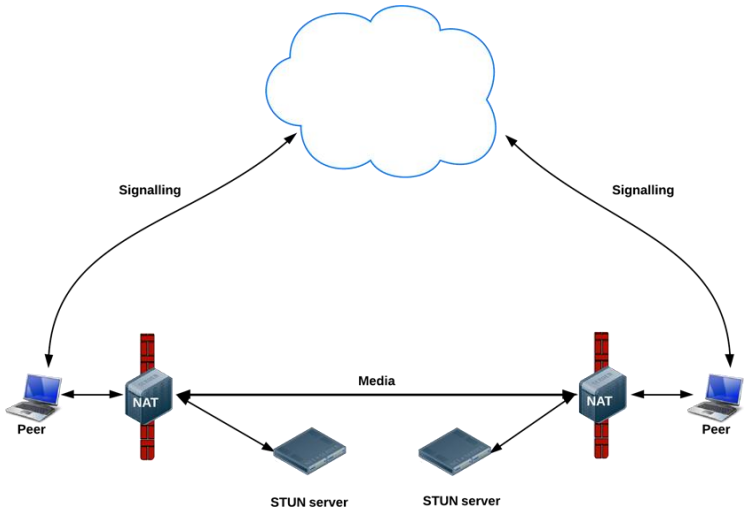
STUN серверите имаат една едноставна задача: понајди ја јавната IP адреса на комуникациската точка (peer), т.е. мапирај ја локалната со јавна IP адреса. Овој процес овозможува крајните точки на комуникација да добијат јавни адреси и да ги испратат меѓу себе преку механизам за сигнализација, за да можат директно да комуницираат.

STUN користи пинг-понг механизам за да најде јавна IP адреса на клиентот, па peer to peer конекцијата може да се воспостави и да се помине низ firewall.

Иницијално ICE се обидува да ги конектира крајните точки директно, со најмалата можна латенција, преку UDP.

STUN серверите немаат многу за памтење, па затоа можат да примат огромен број на барања.

Многу WEBRTC повици реализираат конекција со STUN: 86 % според WebRTCstats.com.



Слика 14. Користење на STUN сервери
Figure 14. Using of STUN servers

2.5.2. TURN Traversal Using Relays around NAT (TURN) protocol (RFC5766)

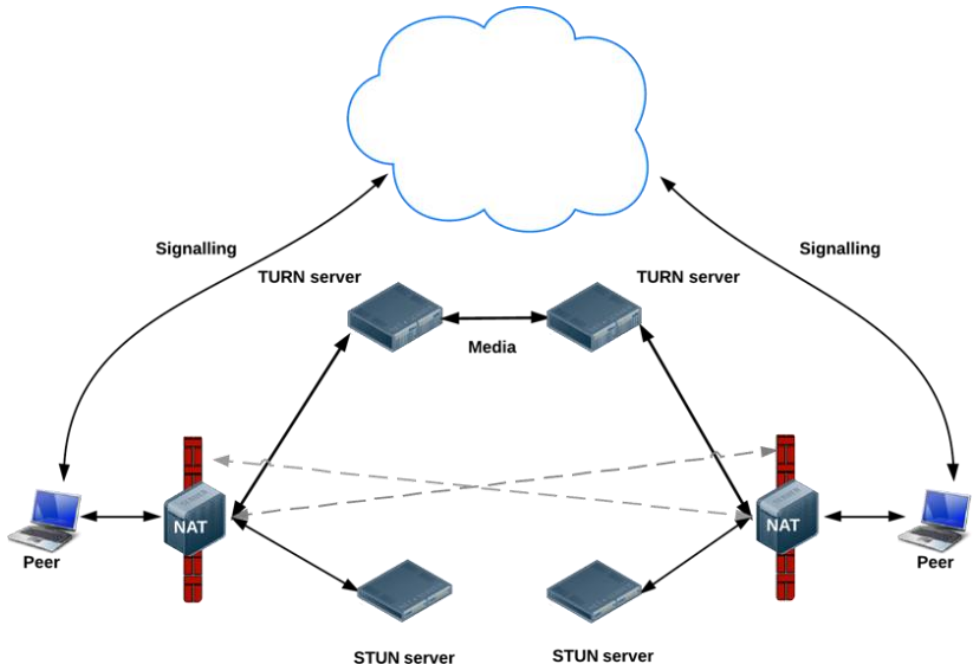
Ако STUN серверот не може да најде пат за директна конекција, поради NAT и firewall, тогаш се користи TURN сервер.

RTCPeerConnection се обидува да воспостави директна комуникација помеѓу крајните уреди преку UDP. Ако тоа не успее, се користи TCP. Ако и тоа не успее, TURN серверите се користат како fallback, препраќајќи податоци помеѓу крајните точки.

Имено, TURN се користи за пренос на аудио и видео меѓу крајните точки, а не за сигнализација.

TURN серверите имаат јавни адреси, па можат да реализираат конекција иако крајните точки се позади firewall или прокси. TURN серверите имаат едноставна задача: да препраќаат стрим.

Овам всушностм води кон конекција која не е p2p, но во некои случаи ова е единствената опција.



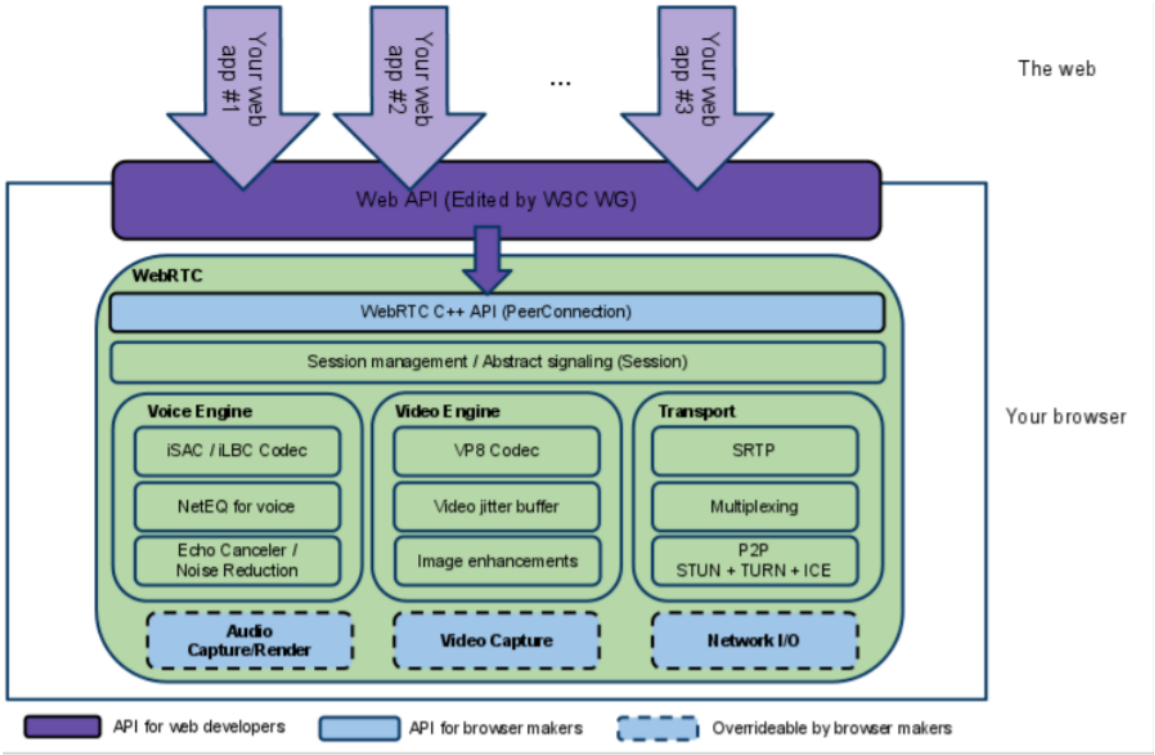
Слика 15. Користење на TURN сервери
Figure 15. Using of TURN servers

2.6. Архитектура на WebRTC

Генералната слика за архитектурата на WebRTC е претставен на слика 16. Архитектурата е составена од два слоја:

1. Слој за снимање и продуцирање аудиовизуелни податоци. Овој слој ќе ги интерисира развивачите на прелистувачи.

2. Web API слој. Овој слој ќе ги интересира развивачите на веб апликации.



Слика 16. WebRTC архитектура
Figure 16. WebRTC architecture

Секој дел од архитектурата претставена на сликата 16 е подетално објаснет подолу:

1. **Your Web App** - претставува веб базирана апликација со аудио и видео можности.
2. **Web API** (Edited by W3C WG) - претставува интерфејс т.е. API коешто ќе биде користено од страна на развивачите на апликации.
3. **WebRTC C++ API (Peer Connection)** - го преставува API слојот кој овозможува лесна имплементација на Web API функциите.

4. **Session Management / Abstract signaling** - ова е слој за управување со сесии и протокол за комуникација.
5. **Voice Engine** - претставува framework за работа со аудио содржината, т.е. пренос на звук од звучната картичка на мрежа.

а. i**SAC / iLBC / Opus iSAC** - претставуваат аудио кодеци за VoIP и стримање на аудио податоци.

б. **NetEQ for Voice** - претставува динамичен jitter бафер. Овој бафер содржи механизам на прикривање на грешки кои настанале (на пример, губење на пакети во мрежа и слично). Има латентност на најниско можно ниво, а квалитет на звук на највисоко можно ниво.

в. **Echo Canceler** - претставува софтвер којшто во реално време го отстранува ехото во аудио сигналот, т.е. има за задача сигналот кој е продуциран на звучникот да не се префрли на микрофонот кој се користи истовремено.

г. **Noise Reduction** - претставува софтвер за намалување на позадинската бучава, т.е. има за задача од сигналот да ги отстрани дополнителните звуци кои доаѓаат во позадина од микрофонот.

6. **Video Engine** - претставува framework за пренос на видео сигнал, од камерата до мрежата, и дополнително од мрежата да се направи приказ на екран.

а. **VP8 Видео кодек** - е open-source видео кодек, кој е познат по својата брзина на декодирање и подобрени перформанси за помала загуба на податоци. Кодот е универзален и лесно може да се имплементира на хардверска платформа, затоа видеоконференциските системи го користат овој кодек.

б. **Video Jitter Buffer** - претставува динамичен jitter бафер за видео. Овој бафер содржи механизам на прикривање на грешки кои настанале (на пример, при губење на паките). Овозможува да ги пополни недостатоците од сигналот за да се овозможи добар квалитет на видеото.

в. **Image enhancements** - елемент за подобрување на слики. На пример, ги отстранува нечистотиите на сликата продуцирани од веб камерата.

7. **Transport** - слој за пренос на податоци

а. **SRTP** – SRTP е екстензија на RTP протоколот со подобрен механизам за безбедност. Обезбедува сигурна енкрипција, автентикација и интегрална верификација на интернет медија податоците, како што се аудио и видео.

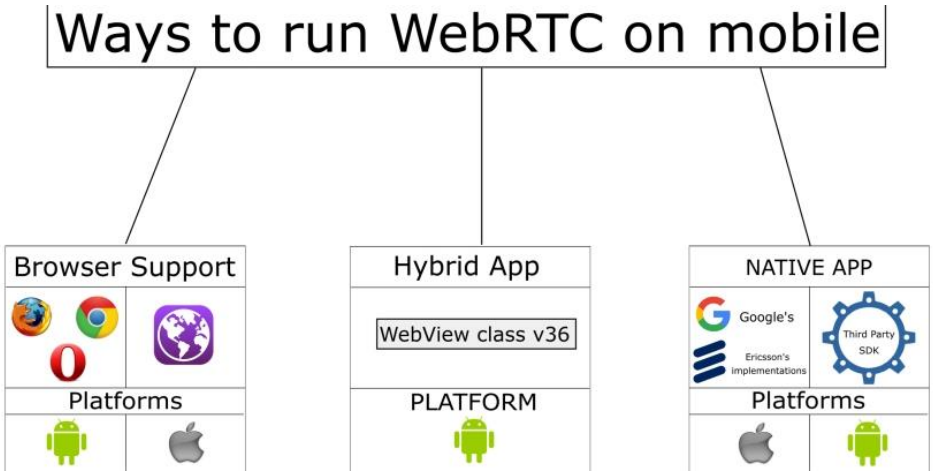
б. **Multiplexing** - е популарна мрежна техника која интегрира повеќе аналогни и дигитални сигнали во еден сигнал кој се пренесува преку комуникациски канал.

в. **STUN/TURN/ICE** - е елемент на WebRTC кој овозможува повиците (аудио и видео) да користат STUN и ICE техники со цел да се воспостави врски преку различни видови на мрежи.

2.7. Модели на WebRTC во мобилниот свет

Постојат 3 модели за WebRTC во мобилниот свет:

- The native browser - Најчесто користениот начин е со native пребарувач. Во овој случај уредот е подготвен да работи со основните конфигурации.
- Browser application - Ова значи да се користи трета апликација (не основниот nativeweb пребарувач) за да се овозможи WebRTC (уште наречена hyrid application, бидејќи се креира хибридна апликација, проширувајќи ја WebView класата, која поддржува WebRTC).
- Native application - Се креира мобилна апликација која ќе ја овозможи WebRTC функционалноста.



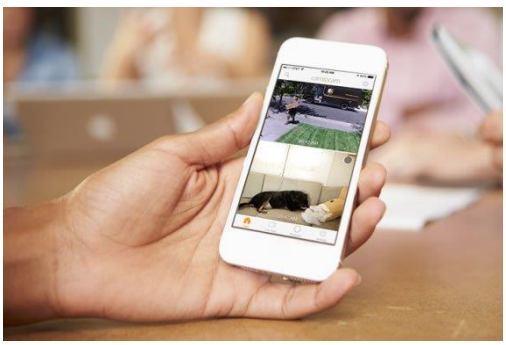
Слика 17. WebRTC мобилни модели
Figure 17. WebRTC mobile models

3. Претходни сродни истражувања и употреба на WebRTC

Постојат многу области (use cases) каде што WEBRTC наоѓа примена. Во прилог се неколку од нив:

3.1. Решенија за мониторинг

WebRTC технологијата е практична за системи за мониторинг, пред сè бидејќи е еден од најбезбедните начини за стриминг на медија. End-to-end енкрипцијата и narative peer-to-peer (p2p) архитектура значат дека медијата која се трансферира не е предмет на надзор или следење од страна на серверот. Исто така, корисниците немаат потреба да ги поставуваат јавно своите интернет мрежи за да користат WiFi безбедноста камера. Девелоперите можат да изградат моќна апликација за мониторинг многу поевтино, користејќи го VP8 видео кодекот. Некои примери на веќе постоечки решенија за мониторинг кои користат WebRTC се: NodeLink, Camio и Amaryllo.



Слика 18. Видео мониторинг
Figure 18. Video monitoring

Постојат повеќе научни студии кои го анализираат проблемот на мониторинг со употреба на WebRTC. Така, на пример, во студијата на Tiberkak et al. [52] предложена е архитектура на иновативен систем за следење и анализа на критични ситуации и настани во домот. Авторите предлагаат интеграција на WebRTC во основата на системот за паметен дом и ги анализираат времињата на доцнење од моментот на појава на несаканиот настан до моментот на добивање на нотификација. Заклучокот кој го изведуваат упатува на фактот дека WebRTC може да се користи како алатка за мониторинг на критични настани во реално време.

3.2. WEBRTC for smart home

Една од најпрофитабилните области за WebRTC и во исто време еден од најралевантните IoT трендови е smart home. На пример, домофони на влезната врата или паметно електронско сандаче кое користи WebRTC за аудио и видео комуникација со веб и мобилните апликации. Резидентите на паметните куќи можат да комуницираат со посетителите и на тој начин да си обезбедат сигурност.

Слика 19. „Паметна куќа“
Figure 19. Smart house

3.3. Онлајн стриминг

Апликации како Rabbit и Airtime сакаат да креираат shared viewing платформи кои работат на сите уреди. Идејата е да се соберат сите луѓе заедно со тоа што им се дозволува да консумираат медија за време на

конверзација во живо. Тие се фокусираат на можноста да пренесуваат медија преку постоечките страници како: YouTube, Netflix, Amazon, Spotify и слични.

Слика 20. Rabbit апликација
Figure 20. Rabbit application

3.4. Корисничка поддршка и сервис

Една од областите каде може да се види вредноста на WebRTC е корисничката поддршка и сервис. Со еден клик видеоповик и поврзување со агент за комуникација во реално време.

1.1. Автомобилство - корисничка поддршка за Uber, Airbnb, Car2Go. Во случај кога ќе се вклучи трепкање на светло за кое не сте сигурни што означуваа, преку систем базиран на WebRTC можете да се конектирате со техничка поддршка и да бидете соодветно упатени.

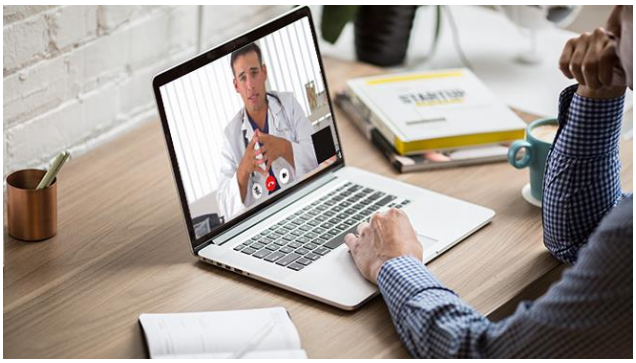
1.2. Продавница за малопродажба со конектирање со remote продавач. Во киоскот купувачите притискаат копче за видеочет со презентерот на брендот во нивниот нативен јазик. Продавачот одговара на прашањата на купувачот и му помага во изборот.

1.3. Онлајн продавници. На пример: Amazon има вклучено поддршка за корисници со клик на копчето „Mayday“ директно на страницата.

1.4. Банкарство - BancSpace е пример за WEBRTC дигатална банкарска комуникациска и колаборациска платформа. Многу банки во Америка имаат додадено видеочет поддршка за корисниците на ATM машините. PeerCDN користи WebRTCData Channel за размена на податоци.

3.5. WEBRTC во здравствена грижа (telehealth/healthcare)

Telehealth е интересен сегмент за работа кога доаѓа во прашање WebRTC. Многу здравствени клиники користат апликации базирани на WebRTC за терапевтски сесии и консултации, со што се намалуваат посетите од пациенти во канцеларија. Ова е добро бидејќи им овозможува на докторите да им дадат приоритет на покритичните и итни пациенти. Исто така, освен релација доктор – пациент, WebRTC е добар и за комуникација на вработените во здравството за консултации во реално време со други доктори.



Слика 21. Telehealth апликација
Figure 21. Telehealth application

На Play Store веќе постојат околу 500 апликации за здравствени грижи кои користат WebRTC и дел од нив имаат голем процент на употреба. Пример е LetsNature, startup од Индија, која нуди апликација за работа во здравството која овозможува закажување на термин со доктор за видеотерапија во реално време. Пациентот комуницира со докторот преку видеоповик во реално време и на тој начин се овозможува брза интервенција и реашавање на проблем.

Овој сервис се очекува дека ќе порасне со профит на 9.3 милијарди долари до 2021 година.

3.6. Онлајн едукација

Релација професор - студент преку предавање, консултации, туторијали, од која било локација во светот.

3.7. Онлајн gaming

Видео и аудио четот во реално време се многу важни за тимски базираните игри, каде што играчите треба да дискутираат и да се консултираат

за стратегијата на играта брзо во реално време, без доцнење на сигналот. WebRTC овде наоѓа примена бидејќи нуди брз трансфер на видео и аудио сигналите, бидејќи не користи централен сервер и притоа овозможува ниска латентност. На пример, играта „The Hobbit: The Battle of the Five Armies“ користи WebRTC.

3.8. Industrial WEBRTC

Индустриските претпријатија се повеќе конзервативни во однос на технологиите и иновациите. Сепак, промени со нови решенија секогаш се потребни и добредојдени. На пример, може да се креира апликација која ќе тригерира видеоповик. Паметна фабрика може да ја користи оваа технологија за да го автоматизира процесот со сензори. На пример, термостат укажува дека машината може да прегрее, па во овој случај може да се вклучи видеокамера за да се направи мониторинг во реално време и да се провери ситуацијата со условите во реално време.

Слика 22. Управување робот со WebRTC апликација
Figure 22. Robot managing with WebRTC application

3.9. Social networks и collaboration

Целта е да се овозможи луѓето да комуницираат, затоа WebRTC доаѓа до употреба. Најпознати апликации се: Google Meet, Google Hangouts, Facebook Messenger, SnapChat, WhatsApp и слично.

WebRTC технологијата овозможува видеоконференциска соработка. Cisco Webex Meetings е систем со пораки кој обезбедува видеоконференција и колаборација користејќи технологија во облаци, мобилни телефони и веб.

4. Софтверско решение

Современото општество не може да се замисли нормално да функционира без лесен и брз пристап до мобилен уред кој истовремено и постојано е со достапна интернет конекција. Уште од најрана возраст навикнуваме да живееме и просперираме со очите вперени во екраните од каде што ги добиваме скоро сите информации (high bandwidth input of info). Затоа, овој магистерски труд е насочен кон креирање прототип на мобилна апликација која овозможува видеокомуникација во реално време. Во прилог е елаборирано објаснувањето.

4.1. Концепт на работа

Концептот на работење е сведен на креирање на соба за комуникација од едниот партиципиент и вклучување на друг партиципиент т.е. краен корисник кој сака да комуницира со првиот.

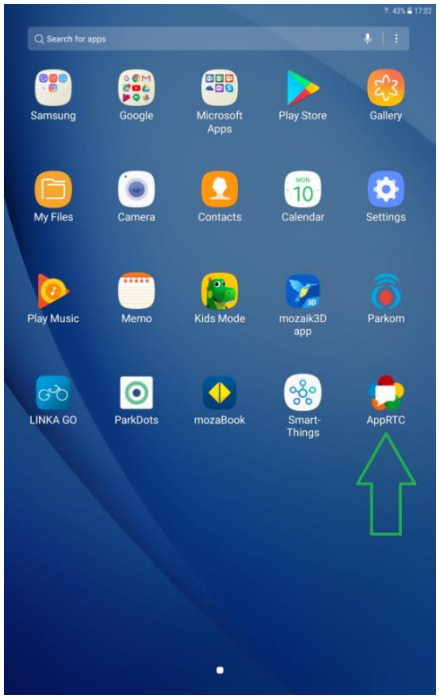
Имено, на двата уреда е потребно да егзистира апликацијата, при што кој било од крајните корисници е потребно да креира уникатно име за соба, кое ќе биде јавно споделено со сите. Друг уред кој сака да воспостави комуникација со првиот го користи ова јавно име за да се вклучи во истата соба и да воспостават аудио и видео комуникација, само со еден клик.

4.2. Ресурси за работа

За креирање на оваа апликација е користено Android Studio како работна околина, верзија 3.1.3, Java за кодот во позадина. Се користи Linux (open SUSE) оперативен систем. Минимални барања за нормално оперативно функционирање по deploy на апликацијатае уредите да имаат поддршка за android (software) и камера и микрофон (hardware).

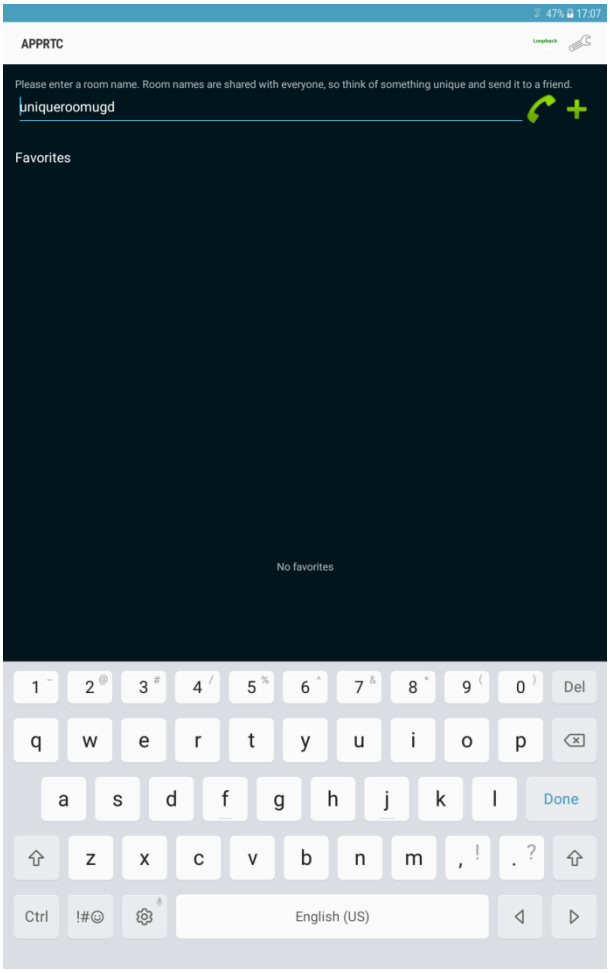
4.3. Кориснички интерфејс

По инсталирањето на апликацијата на андроид уредите, до истата се пристапува на стандарден начин на соодветната икона (слика 23).



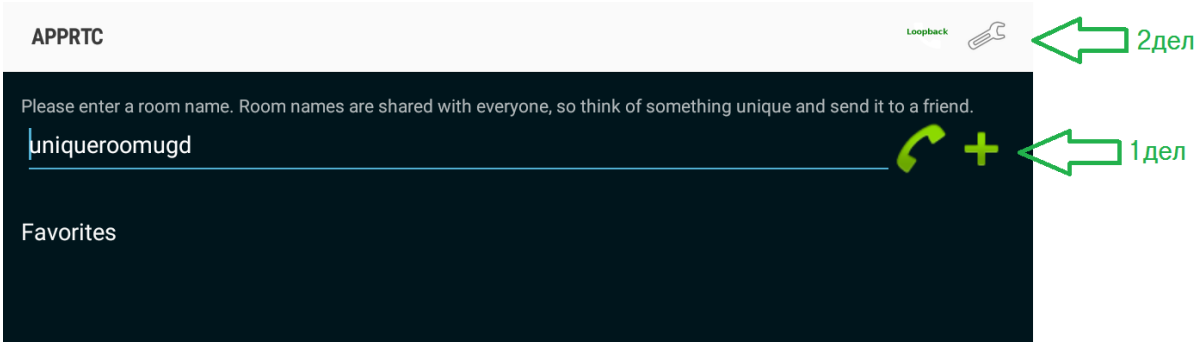
Слика 23. Пристап до апликацијата за видеоповик
Figure 23. Access the video call application

Водејќи се од аспект на едноставност, дизајнот на корисничкиот интерфејс е лесен за маневрирање (слика 24).



Слика 24. Почетен кориснички интерфејс на апликацијата за видеоповик
Figure 24. The video call application's initial user interface

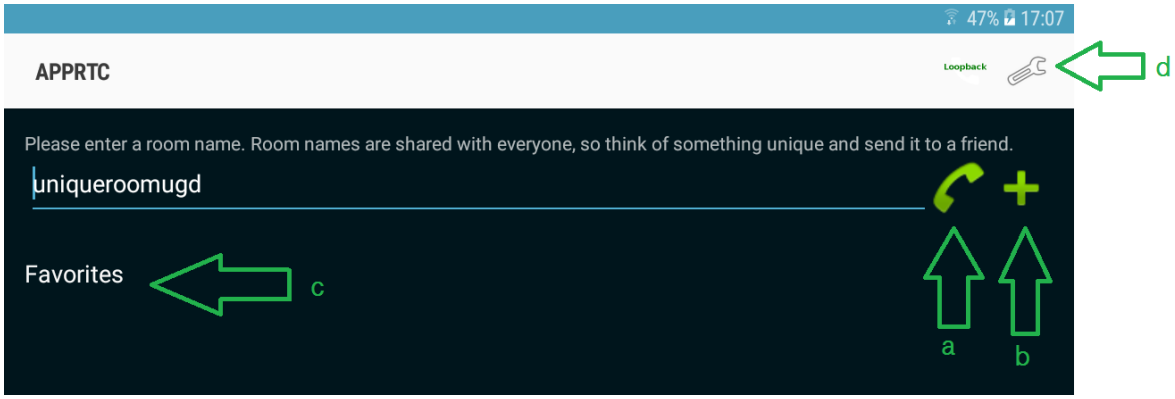
Генерално, почетниот интерфејс е составен од два дела (слика 25): едниот дел се однесува на воспоставување комуникација и другиот дел се однесува на одредени поставки/нагодувања (својства/опции).



Слика 25. Поделба на кориснички интерфејс на апликацијата за видеоповик
Figure 25. Separation of user interface application for video calling

Првиот дел има простор за креирање на име на соба и копче за повик (слика 26, под a). Дополнително има опција за зачувување на името на соба (слика 26, под b), поставувајќи ја во делот „Омилени“ (слика 26, под c).

До вториот дел, кој се однесува за особини и нагодувања се пристапува со клик на копчето во горниот десен агол на апликацијата (слика 26, под d).

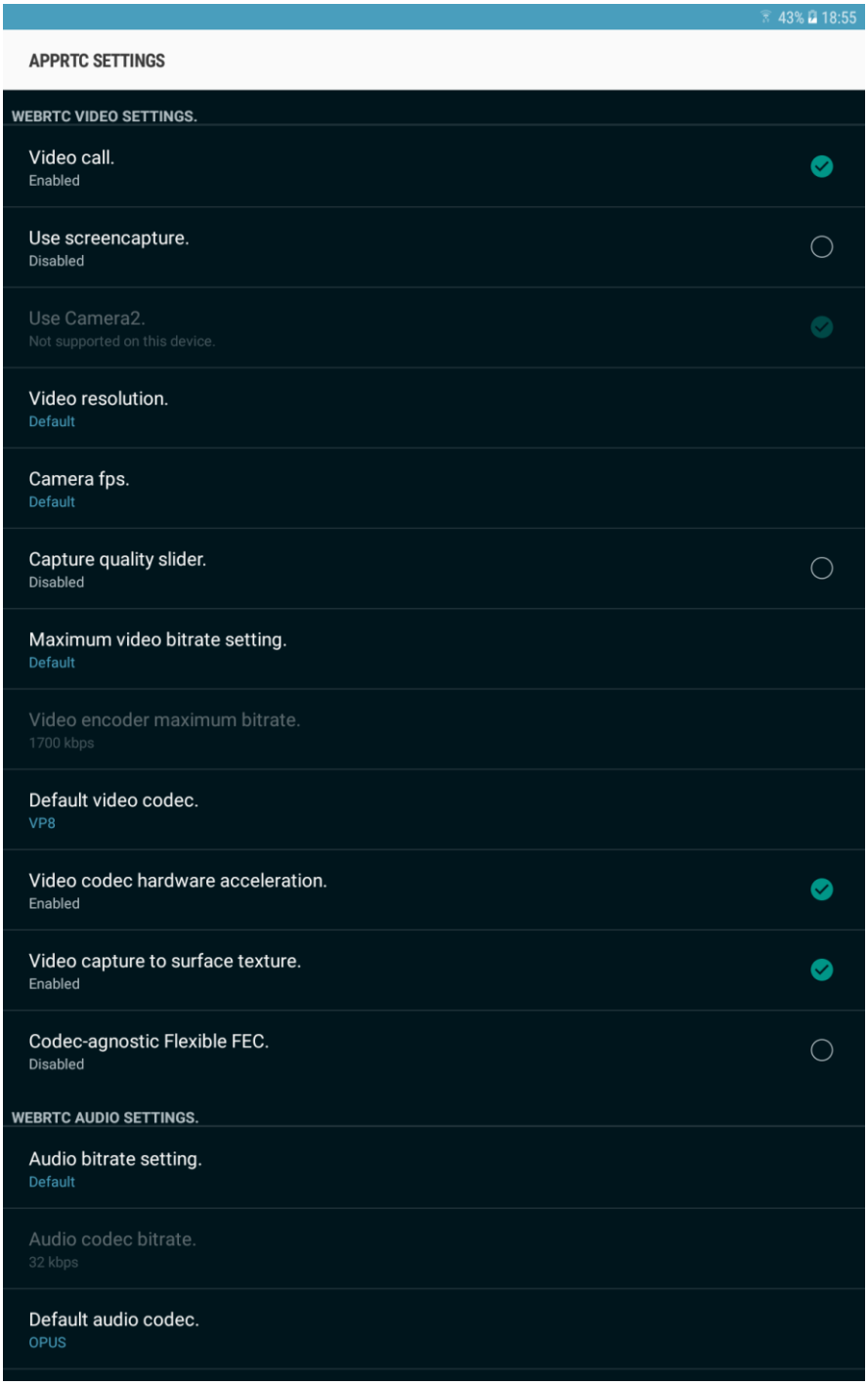


Слика 26. Состав на кориснички интерфејс на апликацијата за видеоповик
Figure 26. Composition of the user interface of the video call application

Во делот за нагодувања има палета од својства за апликацијата и тоа:

- Video call – Вредности: Enabled и Disabled
- Use screencapture - Вредности: Enabled и Disabled. Овозможува споделување на екран.
- Use Camera 2 - Вредности: Enabled и Disabled
- Video resolution - Вредности: Default, 4K, Full HD, HD, VGA, QVGA.
- Camera fps – Вредности: Default, 30 fps, 15 fps.
- Capture quality slider – Вредности: Enabled и Disabled
- Maximum video bitrate setting – Вредности: Enabled и Disabled
- Video encoder maximum bitrate. – Основна вредност е 1700 kbps. Не е достапна за едитирање.
- Default video codec. – Вредности: VP8, VP9, H264 Baseline, H264 Hight
- Video codec hardware acceleration – Вредности: Enabled и Disabled
- Video capture to surface texture – Вредности: Enabled и Disabled
- Codec-agnostic Flexible FEC – Вредности: Enabled и Disabled
- Audio bitrate setting – Вредности: Default и Manual

- Audio codec bitrate – Основна вредност е 32 kbps. Не е достапна за едитирање.
- Default audio codec – Вредности: OPUS, ISAC
- Disable audio processing – Вредности: Enabled и Disabled
- Create aecdump – Вредности: Enabled и Disabled
- Use OpenSL ES for audio playback – Вредности: Enabled и Disabled
- Disable hardware AEC – Вредности: Enabled и Disabled
- Disable hardware AGC – Вредности: Enabled и Disabled
- Disable hardware NS – Вредности: Enabled и Disabled
- Enable level control – Вредности: Enabled и Disabled
- Disable WebRTC AGC and HPF – Вредности: Enabled и Disabled
- Speakerphone – Вредности: Auto (proximity sensor), Enabled и Disabled.
- Enable datachannel – Вредности: Enabled и Disabled
- Order messages – Вредности: Enabled и Disabled
- Subprotocol – Вредност од страна на корисникот
- Negotiated – Вредности: Enabled и Disabled
- Max delay to retransmit – Основна вредост: -1
- Max attempts to retransmit – Основна вредост: -1
- Data id – Основна вредост: -1
- Room server URL – Вредност: https://appr.tc
- Display call statistics – Вредности: Enabled и Disabled
- Debug performance tracing – Вредности: Enabled и Disabled



Слика 27. Дел од опциите за нагодувања на апликацијата
Figure 27.Some setup application’s options

4.4. Screen share (споделување на екран)

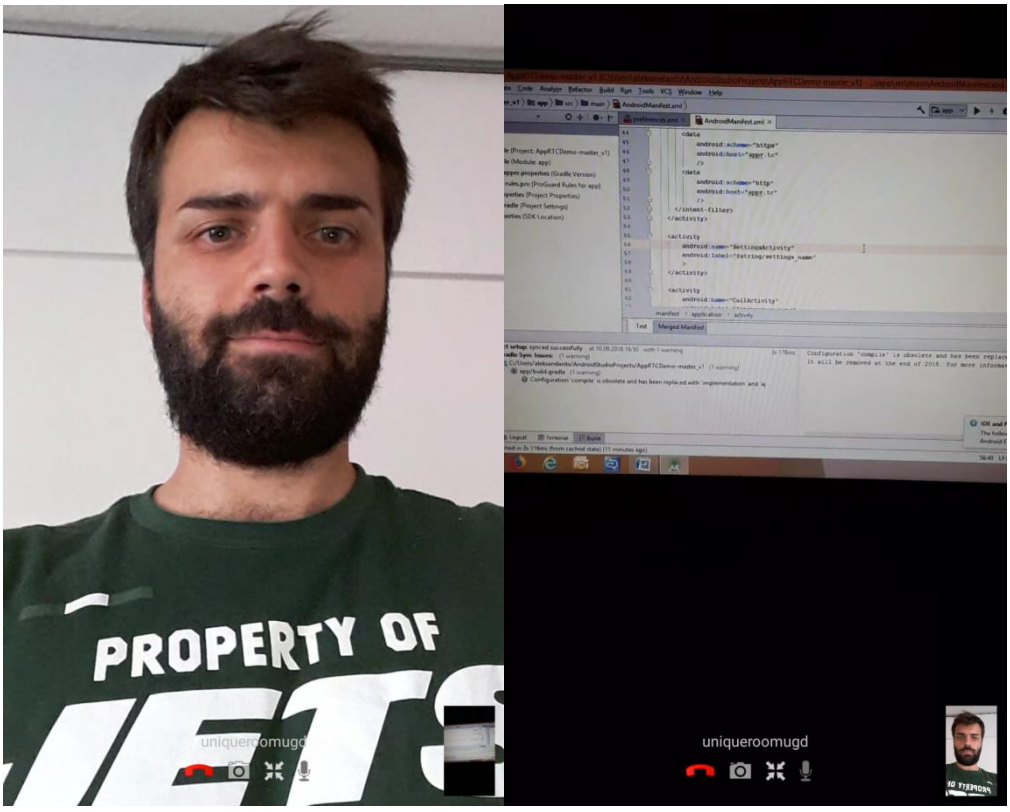
Screen share е особина овозможена преку опцијата Use screen capture од делот за нагодувања на апликацијата. Ова особина овозможува емитирање на содржината на екранот на едниот корисник на екранот на другиот корисник. Оваа функционалност на апликацијата е многу практична при процесот на

соработка, бидејќи овозможува визуелен преглед и можност за поефикасен и ефективен начин за корисничка помош, презентации, за одредени надгледувања и слично.

Аудио и видео сервисот надополнети со опцијата за Share screen се докажале како многу корисни во областа на корисничка/техничка поддршка во реално време. Имено доколку клиентот бара помош од операторот, со оваа опција операторот има можност да го упати и надгледува клиентот во реално време и на тој начин побрзо да го реши проблемот.

4.5. Интерфејс за видеоповик

Откако ќе се воспостави видео контакт помеѓу двата уреда, корисничкиот интерфејс изгледа како на слика 28. Прикажани се името на собата и неколку опции. Од понудените опции се вклучени: исклучување на видеоповикот, промена на предна/задна камера, full screen и Mute (безгласно).



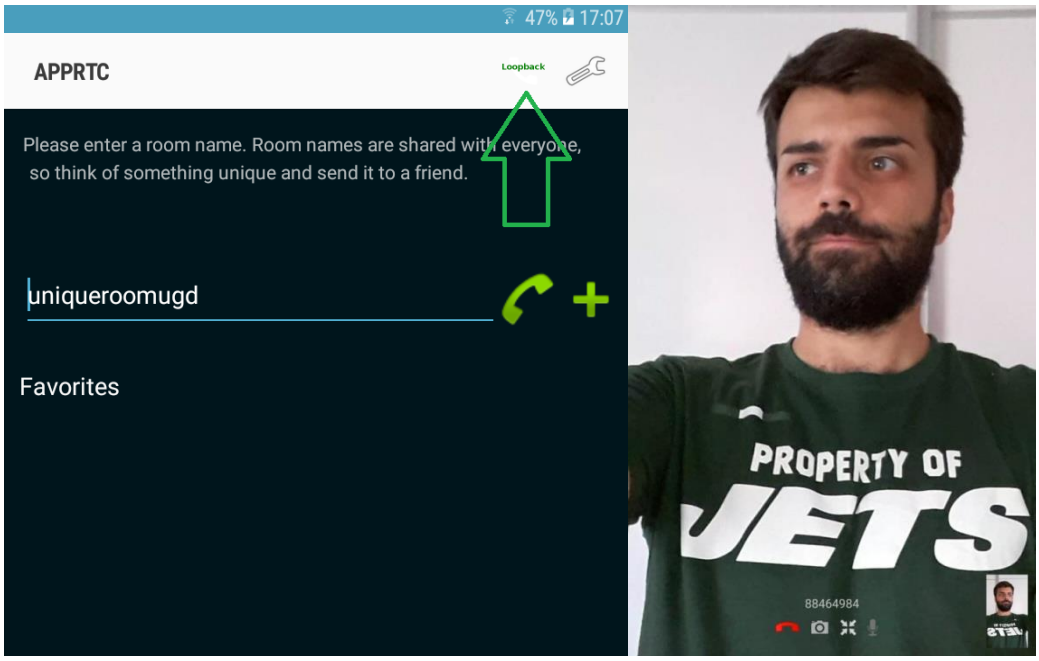
a) b)

Слика 28. Интерфејс при видеоповик, a) интерфејс на PeerA, b) интерфејс на PeerB

Figure 28. Video call interface, a) Interface for PeerA, b) Interface for PeerB

4.6. Run loopback test execute

Loopbacke опцијата на апликацијата поставена во десниот горен агол претставува можност за тестирање на камерата и микрофонот на корисникот. Интерфејсот на видеоповикот при loopback ги има истите опции како нормален видеоповик: исклучување на повикот, промена на камерата, full screen и mute.

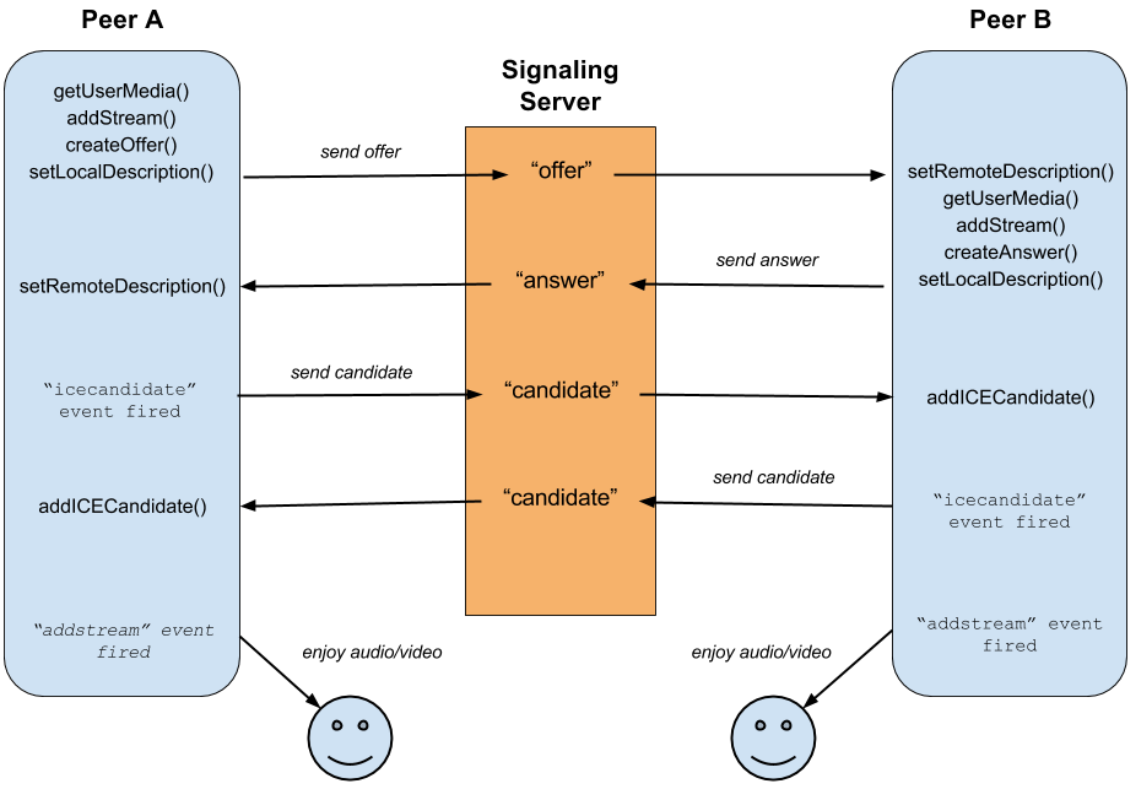


Слика 29. Loopback опција
Figure 29. Loopback option

4.7. Дијаграм на backend активност

На слика 30 е прикажан дијаграмот на активности кои се случуваат во позадина на апликацијата со цел поврзување на двата крајни корисници и воспоставување на аудио и видеоповик.

WebRTC клиентите (попознати како peers) преку процесот на сигнализација се координираат и е потребно да ги разменат информациите за мултимедијални методи, како што се резолуција, пропусен опсег, кодеци и типови на мултимедија. Процесот на сигнализација за размена на конфигурациски информации се реализира со offer (понуда) и answer (одговор) методи со помош на Session Description Protocol (SDP).



Слика 30. Дијаграм на backend активности
Figure 30.Diagram of backend activities

WebRTC користи RTC PeerConnectionAPI за да го сетира видео стримингот помеѓу клиентите (peers):

1. PeerA креира RTCPeerConnection објект.
2. PeerA го стартува createOffer() методот. Со што како одговор се добива RTCSessionDescription т.е. опис за локалната сесија на PeerA. Ако е успешно, тогаш PeerA самостојно ја сетира локалната сесија преку setLocalDescription() и истата ја испраќа на PeerB преку нивниот канал за сигнализација. Притоа RTCPeerConnection не овозможува испраќање на детали за кандидатите сè додека не се нагоди setLocalDescription().
3. PeerB ја сетира кај него оваа сесија како remote description, користејќи го методот setRemoteDescription().
4. PeerB го стартува createAnswer() методот, поставувајќи ја сесијата која ја добил од PeerA како параметар, со цел неговата локална сесија да се изгенерира компатибилна со онаа на PeerA. CreateAnswer() препраќа RTCSessionDescription т.е. опис за локалната сесија на PeerB. Локалната сесија на PeerB се препраќа на PeerA.

5. Откако PeerA ќе ја добие local session description на PeerB, истата ја поставува како remote description преку методот setremoteDescription()

Исто така, при процесот на сигнализација е потребно да се разменат мрежни информации. Имено, фразата 'finding candidates' укажува на процесот на наоѓање на интерфејс и порт со помош на ICE framework:

6. PeerA go повикува методот onicecandidate();
7. Кога PeerB добива кандидатска порака од PeerA, се повикува addIceCandidate() со цел да се додаде кандидат преку remote peer description;
8. Со ова успешно се воспоставува видео конекцијата.

4.8. Backend карактеристики на апликацијата

Во прилог се опишани некои карактеристики кои се важни за апликацијата и се однесуваат на backend процесот:

- Главни активности на апликацијата се:

    1. ConnectActivity – главниот екран – ги прикажува конектираните корисници. Ова е првата активност која се повикува при стартување на апликацијата и истата се справува со иницијални сетинзи каде што корисникот сетира која соба ќе ја користи.

    2. CallActivity – екранот кој го има видеото и неговите контроли.

- Целата websocket комуникација се реализира во:

    1. Websocket Channel Client - креира, конектира, регистрира и затвора websocket за Signaling Server;

    2. Websocket RTC Client – прима WebRTC-сигнални пораки и се справува со нив соодветно;

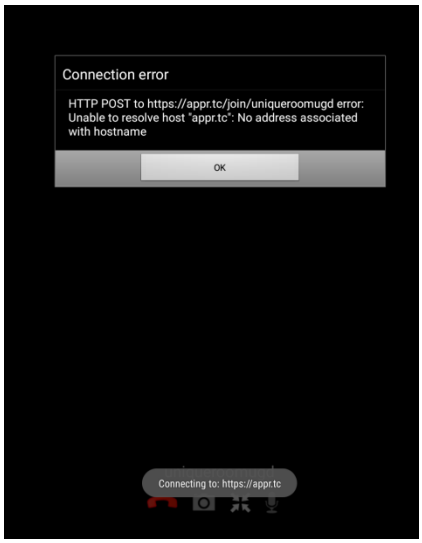
- WebRTC peerconnection е реализирана во PeerConnectionClient;

- Се користи пакетот org.appspot.apprtc и https://appr.tc/ како сервер.

- Во AndroidManifest.xml се доделуваат привилегии за да може WebRTC да работи:

```
uses-feature android:name="android.hardware.camera"/>
<uses-feature android:name="android.hardware.camera.autofocus"/>
<uses-feature
android:glEsVersion="0x00020000"
android:required="true"/>
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission
android:name="android.permission.CHANGE_NETWORK_STATE"/>
<uses-permission
android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

- Уредот кој треба да комуницира мора да има интернет пристап, во спротивен случај се појавува контролна порака како на сликата:



Слика 31. Контролна порака за интернет конекција
Figure31. Control message for Internet connection

4.9. Backend чекори при креирање видео и аудио повик

1. Процесот на конекција започнува со внес на име на соба. Во backend се повикува CallActivity.java со своите методи. Воведена е валидација за да не дозволува null вредност:

```
String roomId = intent.getStringExtra(EXTRA_ROOMID);
if (roomId == null || roomId.length() == 0) {
  logAndToast(getString(R.string.missing_url));
  setResult(RESULT_CANCELED);
  finish();
return;
}
```

   Ако не се внесе никаква вредност за име на соба се прикажува порака: „ERROR: Missing URL to connect to“.

2. Доколку е валидно, се прават следни проверки во однос на тоа дали е сетирана оцијата за screen shareили не. Во случај да е сетирана оваа опција се повикува методот startScreenCapture(), а ако не е селектирана се повикува startCall() методот.

   a) Функција за screen share

```
 private void startScreenCapture() {
   MediaProjectionManager mediaProjectionManager =
       (MediaProjectionManager) getApplication().getSystemService(
           Context.MEDIA_PROJECTION_SERVICE);
   startActivityForResult(
       mediaProjectionManager.createScreenCaptureIntent(),
 CAPTURE_PERMISSION_REQUEST_CODE);
 }
```

   b) Функција startCall()

```
private void startCall() {
if (appRtcClient == null) {
return;
  }
callStartedTimeMs = System.currentTimeMillis();

// Започниконекција со соба.
logAndToast(getString(R.string.connecting_to,
roomConnectionParameters.roomUrl));
appRtcClient.connectToRoom(roomConnectionParameters);

// Креирај audio manager кој ке се погрижи заснимање аудио,
  // аудиомод, аудио уредиенумерацијаитн.
audioManager = AppRTCAudioManager.create(getApplicationContext());
// Зачувај ги посточките audio нагодувања и промени audio модза
  // MODE_IN_COMMUNICATION занајдобриможни VoIP перформанси.
Log.d(TAG, "Starting the audio manager...");
audioManager.start(new AudioManagerEvents() {
// Овојметодќе се повикува секогаш кога ќе има достапно аудио
```

```
промени.

@Override
public void onAudioDeviceChanged(
        AudioDevice audioDevice, Set<AudioDevice>
availableAudioDevices) {
      onAudioManagerDevicesChanged(audioDevice,
availableAudioDevices);
    }
  });
}
```

3. Во startCall() методот се повикуваат неколку други методи, кои доведуваат до повик на createPeerConnection() со што започнува процесот на конекција за повик.

```
private void onConnectedToRoomInternal(final
SignalingParametersparams)
{
final long delta = System.currentTimeMillis() -callStartedTimeMs;
signalingParameters = params;
//Приказ на порака за креирање peer connection со одложување изразено
во милисекунди
logAndToast("Creating peer connection, delay=" + delta + "ms");
VideoCapturer videoCapturer =null;
if (peerConnectionParameters.videoCallEnabled) {
    videoCapturer = createVideoCapturer();
  }

//Повикување на createPeerConnection методот
peerConnectionClient.createPeerConnection(rootEglBase.getEglBaseConte
xt(),localProxyRenderer,remoteRenderers, videoCapturer,
signalingParameters);

//Ако станува збор за оној кој го иницира повикот (т.е кој ја
стартува собата) се повикува овој дел
if (signalingParameters.initiator)
 {
//Приказ на порака
logAndToast("Creating OFFER...");

  //Испраќање offer
peerConnectionClient.createOffer();
  }
//Ако станува збор за оној кој го прифаќа повикот (т.е кој се
приклучува на собата) се повикува овој дел
else {
if (params.offerSdp != null)
 {
peerConnectionClient.setRemoteDescription(params.offerSdp);
//Приказ на порака
logAndToast("Creating ANSWER...");

       //Испраќање одговор
peerConnectionClient.createAnswer();
}
if (params.iceCandidates != null)
```

```
        {
        // Додавање remote ICE кандидати од соба.
        for (IceCandidate iceCandidate : params.iceCandidates) {
        peerConnectionClient.addRemoteIceCandidate(iceCandidate);}
          }
            }
        }
```

4. Кога createPeerConnection() е повикана прекуметодот startCall() се реализира иницирање на повик со креирање на offer (понуда) кој ја содржи локалната сесија и кој се испраќа до peer кој се повикува. На корисничкиот интерфејс се прикажува порака „CREATING OFFER…“.

5. Порака до далечинскиот peer е со серијализирана содржина од локалниот SessionDescription. Овој процес е овозможен од OnLocalDescription() методот од PeerConnectionEvents. На корисничкиот интерфејс се прикажува порака „Sending OFFER“.

```
public void onLocalDescription(final SessionDescription sdp)
{
final long delta = System.currentTimeMillis() - callStartedTimeMs;

        runOnUiThread(new Runnable()
        {
        @Override
        public void run()
        {
        if (appRtcClient != null)
         {
        //Приказ на порака на екран: Sendng offer
                logAndToast("Sending " + sdp.type + ", delay=" + delta +
        "ms");
        if (signalingParameters.initiator) {
        appRtcClient.sendOfferSdp(sdp);}
        else {
        appRtcClient.sendAnswerSdp(sdp); }
              }
        if (peerConnectionParameters.videoMaxBitrate >0)
         {
                Log.d(TAG, "Set video maximum bitrate: " +
        peerConnectionParameters.videoMaxBitrate);

        peerConnectionClient.setVideoMaxBitrate(
        peerConnectionParameters.videoMaxBitrate);
              }
            }
         });
 }
```

6. Откако RTCPeerConnection е успешно креиран со createPeerConnection(), се извршува onIceCandidate() callback функцијата, која како параметар носи информации за кандидатот како што доаѓа т.е. се приклучува на собата.

```
public void onIceCandidate(final IceCandidate candidate) {
  runOnUiThread(new Runnable() {
@Override
public void run() {
if (appRtcClient != null) {
//Се испрќа ICE кандидат до другиот партиципиент
appRtcClient.sendLocalIceCandidate(candidate);
      }
    }
  });
}

public void sendLocalIceCandidate(final IceCandidate candidate) {
handler.post(new Runnable() {
@Override
public void run() {
      JSONObject json = new JSONObject();
jsonPut(json, "type", "candidate");
jsonPut(json, "label", candidate.sdpMLineIndex);
jsonPut(json, "id", candidate.sdpMid);
jsonPut(json, "candidate", candidate.sdp);
if (initiator) {
if (roomState != ConnectionState.CONNECTED) {
          reportError("Sending ICE candidate in non connected
state.");
return;
        }
        sendPostMessage(MessageType.MESSAGE, messageUrl,
json.toString());
if (connectionParameters.loopback) {
events.onRemoteIceCandidate(candidate);
        }
      } else {
wsClient.send(json.toString());
      }
    }
  });
}
```

7. Пораки од клиентот до серверот се прекаат преку sendPostMessage() методот

```
// Send SDP or ICE candidate to a room server.
private void sendPostMessage(
final MessageType messageType, final String url, final String
message) {
  String logInfo = url;
if (message != null) {
    logInfo += ". Message: " + message;
  }
```

```
  Log.d(TAG, "C->GAE: " + logInfo);
  AsyncHttpURLConnection httpConnection =
new AsyncHttpURLConnection("POST", url, message, new
AsyncHttpEvents() {
@Override
public void onHttpError(String errorMessage) {
          reportError("GAE POST error: " + errorMessage);
        }

@Override
public void onHttpComplete(String response) {
if (messageType == MessageType.MESSAGE) {
try {
              JSONObject roomJson = new JSONObject(response);
              String result = roomJson.getString("result");
if (!result.equals("SUCCESS")) {
                reportError("GAE POST error: " + result);
              }
            } catch (JSONException e) {
              reportError("GAE POST JSON error: " + e.toString());
            }
          }
        }
      });
  httpConnection.send();
}
```

8. Пораките од серверот до клиентот се испраќаат со помош на Google App Engine Channel API.

9. Ако пораката е answer на offer кој бил претходно иницран, тогаш RTCPeerConnection го сетира remote SessionDescription, преку методот onRemoteDescription() и креира иницирање на одговор за повикот, при што на кориснички интерфејс (нареег кој се приклучува кон собата) се прикажува порака „Creating ANSWER…“.

```
public void onRemoteDescription(final SessionDescription sdp) {
final long delta = System.currentTimeMillis() - callStartedTimeMs;
  runOnUiThread(new Runnable() {
@Override
public void run() {
if (peerConnectionClient == null) {
        Log.e(TAG, "Received remote SDP for non-initilized peer
connection.");
return;
      }
      logAndToast("Received remote " + sdp.type + ", delay=" + delta
+ "ms");
peerConnectionClient.setRemoteDescription(sdp);
if (!signalingParameters.initiator) {
        logAndToast("Creating ANSWER...");
// Create answer. Answer SDP will be sent to offering client in
        // PeerConnectionEvents.onLocalDescription event.
peerConnectionClient.createAnswer();
```

```
        }
      }
    });
}
```

10. Одговорот се испраќа до иницијаторот на повикот со што се прикажува порака на корисничкиот интерфејс: „Sending answer“.

11. Со испраќањето offer и answer можеме да кажеме дека Peer меѓусебно си имаат испратено информации и се поврзани, со што real-time комуникацијата може да започне. При воспоставување на конекцијата и стартување на видеоповикот на корисничкиот интерфејс на двата peer се покажува порака „Ice connected“, иницирана од onIceConnected() методот.

```
public void onIceConnected() {
final long delta = System.currentTimeMillis() - callStartedTimeMs;
  runOnUiThread(new Runnable() {
@Overriden
public void run() {
       logAndToast("ICE connected, delay=" + delta + "ms");
iceConnected = true;
       callConnected();
     }
  });
}
```

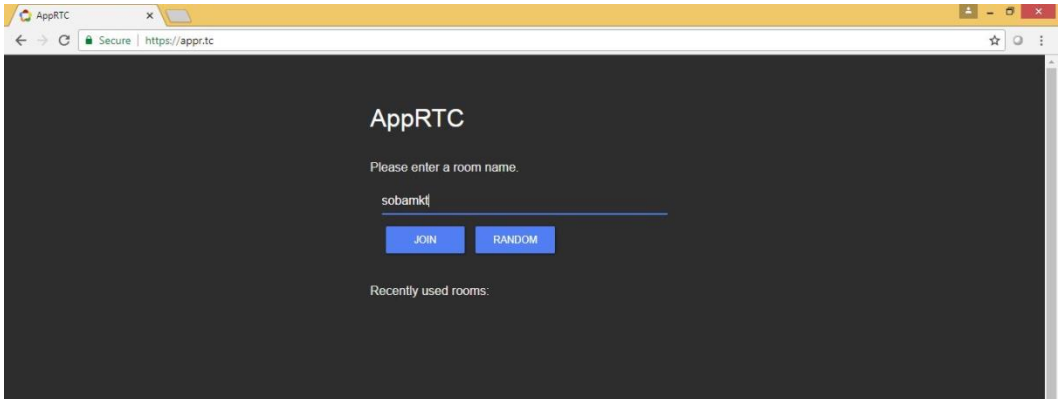
4.10. Error handling

Постојат контролни пораки кои се прикажуваат при интеракцијата:

- нема интернет,
- погрешен url,
- полна соба,
- други грешки.

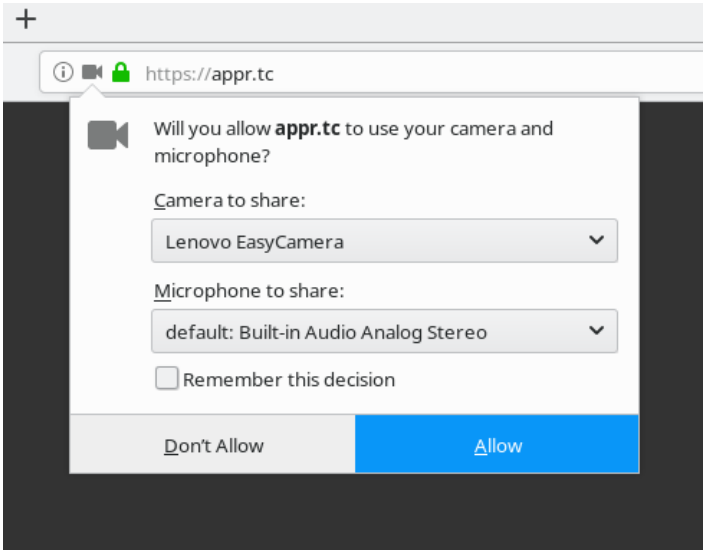
4.11. Комуникација помеѓу веб пребарувачи

Имплементираната WebRTC апликација е приспособена и за работа помеѓу веб пребарувачи. Принципот на работа е ист. PeerA креира соба на линкот https://appr.tc/ чека да се приклучу PeerB (слика 32).



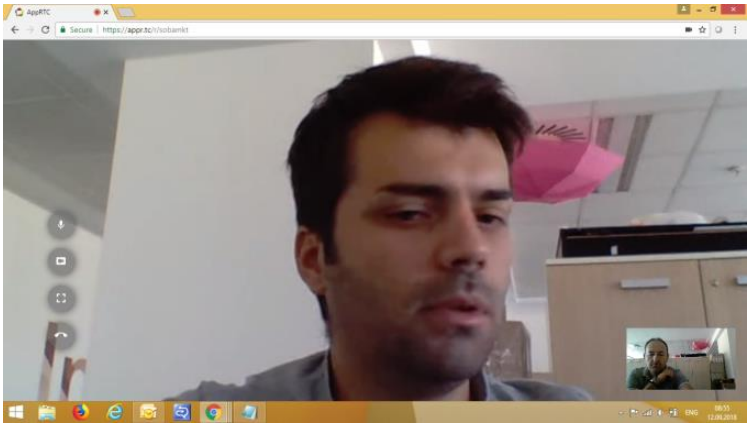
Слика 32. WebRTC комуникација помеѓу веб пребарувачи
Figure 32. WebRTC communication between veb browsers

Притоа веб пребарувачот побарува одобрување за користење на камера и микрафон од уредот, како на слика 33.

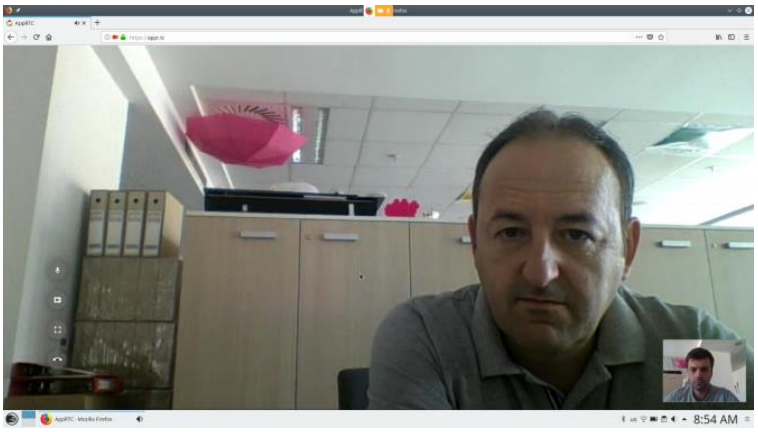


Слика 33. Порака за одобрување користење камера и микрофон
Figure 33. Message for allowing usage of camera and mifrofon

Кога PeerB ќе се приклучи на истата соба, комуникацијата се воспоставува и видеоповикот е успешен (слика 34).



a) PeerA



b) PeerB

Слика 34. WebRTC видеоповик преку пребарувач, a) PeerA b) PeerB
Figure 34. WebRTC video call trought web browser, a) PeerA b) PeerB

5. Евалуација на предложеното софтверско решение

За тестирање на апликацијата елаборирана погоре се користени следните уреди:

Табела 1. Уреди користени при тестирање
Table 1. Devices used in testing

Тип на уред	Назив на уред	Верзија на оперативен систем	API ниво
Мобилен уред	SONY XPERIA XA	Android 7.0 Nougat	24
Мобилен уред	SamsungGalaxy S4	Android 5.0.1 Lollipop	21
Таблет	Samsung Galaxy Tab A SM - TM580	Android 7.0 Nougat	24

Реализирани се повеќе сценарија, со цел тестирање на одредени специфики на апликацијата:

1. Тест сценарио – Кориснички интерфејс на различни уреди
2. Тест сценарио – Конекција без интернет
3. Тест сценарио – Користење на различни типови на безжични мрежи
4. Тест сценарио – Комуникација меѓу исти типови на уреди
5. Тест сценарио – Комуникација меѓу различни типови на уреди
6. Тест сценарио – Споделување екран (Share desktop)
7. Тест сценарио – Low battery behavior
8. Тест сценарио – Low storage space
9. Тест сценарио– CPU перформанси.

Подолу во табелите од табела 2 до табела 10 подетално се објаснети сите тест сценарија, како и резултатот и заклучокот од тестирањето.

Табела 2. Број на тест сценарио 1
Table 2. Number on test scenario 1

Назив на тест сценарио	Кориснички интерфејс на различни уреди
Цел	Проверка дали изгледот на апликацијата е во ред на

	различни уреди, со различен тип на екран во однос на димензии и резолуција
Резултат/статус	Успешен приказ/во ред
Заклучок	Постои голема приспособливост и скалабилност на корисничкиот интерфејс на апликацијата за различни уреди

Табела 3. Број на тест сценарио 2
Table 3. Number on test scenario 2

Назив на тест сценарио	Конекција без интернет
Цел	Проверка дали е возможно поврзување без вклучен интернет. За комуникација е потребна интернет конекција, во спротивно се прикажува соодветна порака
Резултат/статус	Успешен приказ на порака/во ред
Заклучок	Апликацијата успешно прикажува порака при недостаток на интернет конекција

Табела 4. Број на тест сценарио 3
Table 4. Number on test scenario 3

Назив на тест сценарио	Користење на различни типови на безжични мрежи
Цел	Воспоставување на конекција меѓу уредите при различни типови на безжични мрежи: WiFi, GSM
Резултат/статус	Успешна конекција и функционирање на апликацијата/во ред
Заклучок	Апликацијата нема ограничување во однос типот на безжичната мрежа. Апликацијата успешно и без проблеми функционира без разлика на типот на безжични мрежи.

Табела 5. Број на тест сценарио 4
Table 5. Number on test scenario 4

Назив на тест сценарио	Комуникација меѓу исти типови на уреди
Цел	Воспоставување на комуникација меѓу исти типови на уреди. На пример: комуникација помеѓу два мобилни уреди или помеѓу два таблети.
Резултат/статус	Успешно функционална апликација/во ред
Заклучок	Апликацијата успешно работи при ваков тип комуникација

Табела 6. Број на тест сценарио 5
Table 6. Number on test scenario 5

Назив на тест сценарио	Комуникација меѓу различни типови на уреди
Цел	Воспоставување на комуникација меѓу различни типови на уреди. На пример: комуникација помеѓу мобилен уред и таблет
Резултат/статус	Успешно функционална апликација/во ред
Заклучок	Апликацијата нема ограничување во однос на типот на уредот. Комуникацијата успешно се воспоставува без разлика на тоа типот на уредот (мобилен телефон, таблет)

Табела 7. Број на тест сценарио 6
Table 7. Number on test scenario 6

Назив на тест сценарио	Споделување екран (Screen share)
Цел	Во случај да е вклучена опцијата за share desktop, да се реализира успешно споделување на екран при комуникацијата
Резултат/статус	Успешна функционалност/во ред
Заклучок	Функционалноста се реализира успешно

Табела 8. Број на тест сценарио 7
Table 8. Number on test scenario 7

Назив на тест сценарио	Low battery behavior
Цел	Воспоставување успешна комуникација при помал процент на батерија
Резултат/статус	Не е во ред
Заклучок	Кога андроид уредот има помал процент на батерија под 5 % видеоповикот не може да се воспостави

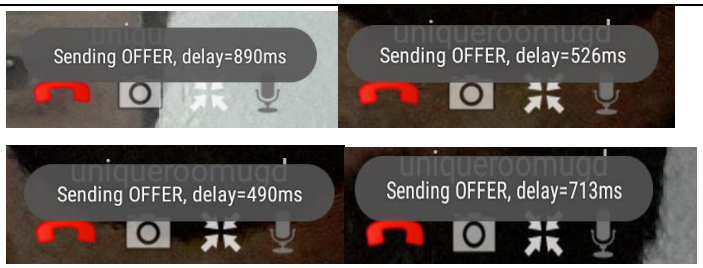
Табела 9. Број на тест сценарио 8
Table 9. Number on test scenario 8

Назив на тест сценарио	Low storage space
Цел	Воспоставување на успешна конекција кога уредот има помала расположлива меморија
Резултат/статус	Успешна функционалност/во ред
Заклучок	Конекцијата и комуникацијата нема ограничување во однос на расположливиот простор на уредот

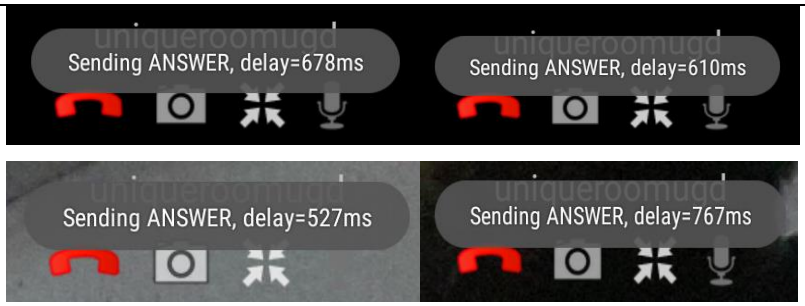
Табела 10. Број на тест сценарио 9
Table 10. Number on test scenario 9

Назив на тест сценарио	CPU перформанси
Цел	Воспоставување на успешна конекција кога уредот има послаби CPU перформанси
Резултат/Статус	Бидејќи WebRTC не поддржуваат уредите кои немаат ARMv7 CPU архитектура, тестирањето е направено на уреди кои имаат ARMv7 или повисока CPU. Согласно со спроведеното тестирање комуникацијата е во ред.
Заклучок	Минимална CPU архитектура е ARMv7

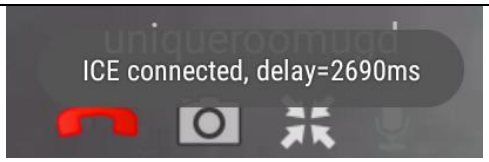
Табела 11. Број на тест сценарио 10
Table 11. Number on test scenario 10

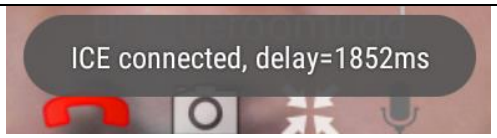
Назив на тест сценарио	Времето потребно за клиентот А да се поврзе на сервер, и да испрати понуда на клиентот В
Цел	Најкратко можно време за поврзување
Резултат/статус	
Заклучок	Времето за поврзување е помало од 1000 ms

Табела 12. Број на тест сценарио 11
Table 12. Number on test scenario 11

Назив на тест сценарио	Времето потребно за клиентот В да се поврзе на сервери да одговори на понудата од клиентот А
Цел	Најкратко можно време за поврзување
Резултат/статус	
Заклучок	Времето за поврзување е помало од 1000 ms

Табела 13. Број на тест сценарио 12
Table 13. Number on test scenario 12

Назив на тест сценарио	Времето потребно за воспоставување на видео сесија (handshake)
Цел	Најкратко можно време за поврзување
Резултат/статус	

	
Заклучок	Времето за поврзување е помало од 3000 ms

5.1. Анализа на користењето на апликацијата за видеоповик

Апликацијата беше презентирана кај повеќе корисници од различна таргет група. Таргет групите беа класифицирани по возраст и област на примена на апликацијата.

Беше објаснет начинот на користење на истата и придобивките од неа. На истите им беше овозможено да имаат интеракција со апликацијата и да ја користат во пракса.

Со цел евалуација на апликацијата и стекнување слика за користеноста и задоволството од истата се спроведени низа интервјуа и е спроведена анкета. Во табела 14 се претставени прашањата за анкета.

Табела 14. Прашања за анкета
Table 14. Questions for a survey

	Прашање	Понудени одговори
1.	Дали сметате дека Ви е потребен ваков тип на апликација за соработка?	Да Не Можеби
2.	Дали апликацијата Ви беше лесна за управување?	Да Не Средно
3.	Колку сте задоволни од корисничкиот интерфејс (изглед) на апликацијата?	Незадоволен Задоволен Многу задоволен
4.	Колку сте задоволни од функционалноста на апликацијата?	Незадоволен Задоволен Многу задоволен
5.	Дали би продолжиле со користење на апликацијата?	Да Не Можеби
6.	Кои се придобивките од ваков тип на апликација?	Отворен тип

7.	Ваше мислење каде најмногу би нашла примена оваа апликација?	Отворен тип
8.	Што дополнително би сакале да вклучува апликацијата?	Отворен тип

5.2. Структура на испитаниците од анкета

Во процесот на евалуација на развиената апликација учествуваа вкупно 30 корисници, кои дадоа свое мислење и искуството од интеракцијата со апликацијата.

Старосна структура:

- Помеѓу 18-50 години: 27
- Над 50 години: 3

Образовна структура

- Средношколци: 2
- Студенти: 12
- Завршено високо образование: 16.

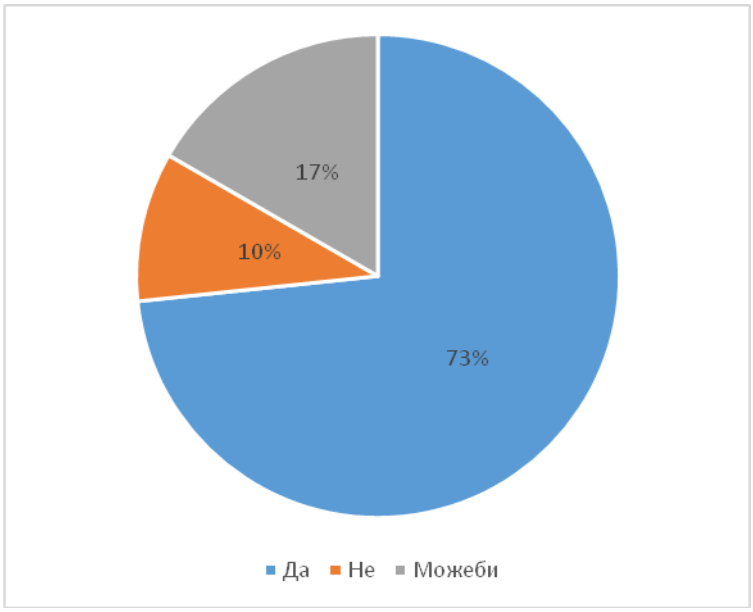
5.3. Резултати и заклучоци од анкета

Резултатите од спроведената анкета се следните:

1. Дали сметате дека Ви е потребен ваков тип на апликација за соработка?

Табела 15. Резултати за прашање 1 од анкетата
Table 15. Results for question 1 of the survey

Одговори	Број на корисници	Процентуален број на корисници изразени
Да	22	73,00%
Не	3	10,00%
Можеби	5	17,00%

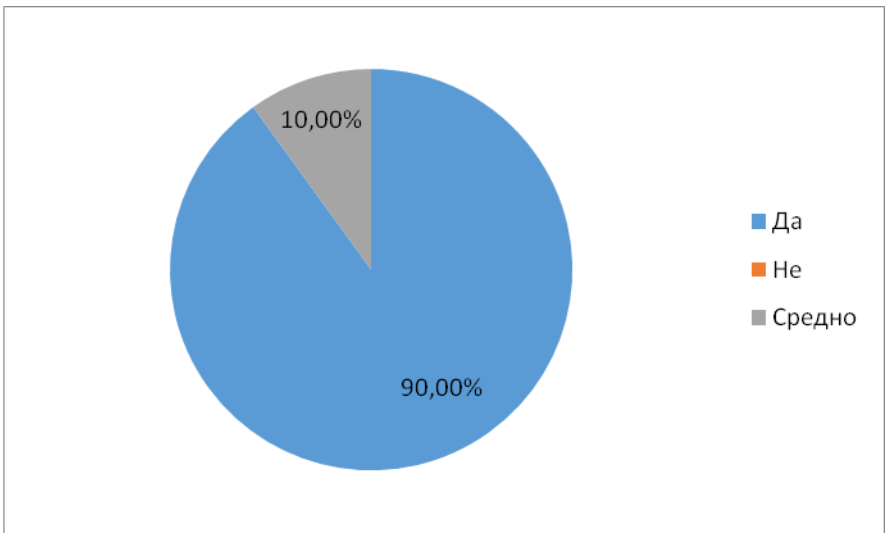


Графикон 1. Процентуален резултат на прашање 1 од анкетата
Graph 1. Percentage result on question 1 of the survey

2. Дали апликацијата Ви беше лесна за управување?

Табела 16. Резултати за прашање 2 од анкетата
Table 16. Results for question 2 of the survey

Одговори	Број на корисници	Процентуален број на корисници изразени
Да	27	90,00%
Не	0	0,00%
Средно	3	10,00%

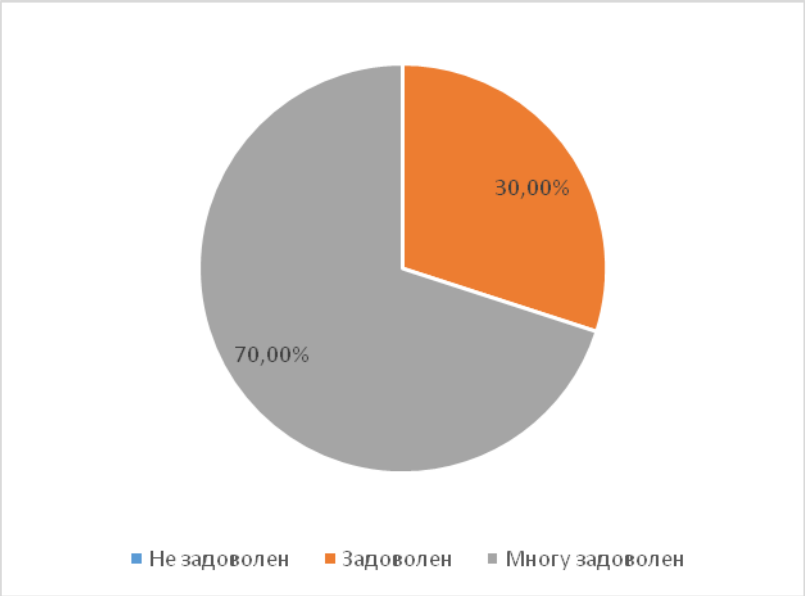


Графикон 2. Процентуален резултат на прашање 2 од анкетата
Graph 2. Percentage result on question 2 of the survey

3. Колку сте задоволни од корисничкиот интерфејс (изглед) на апликацијата?

Табела 17. Резултати за прашање 3 од анкетата
Table 17. Results for question 3 of the survey

Одговори	Број на корисници	Процентуален број на корисници изразени
Не задоволен	0	0,00%
Задоволен	9	30,00%
Многу задоволен	21	70,00%

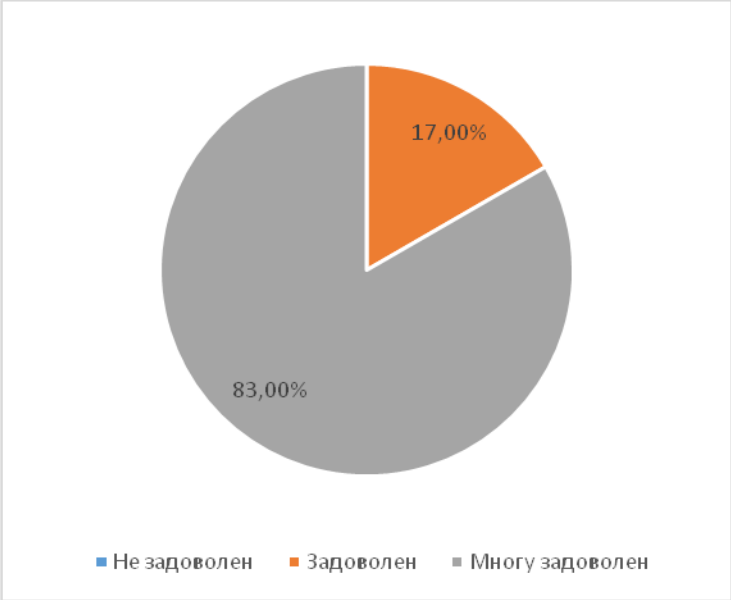


Графикон 3. Процентуален резултат на прашање 3 од анкетата
Graph 3. Percentage result on question 3 of the survey

4. Колку сте задоволни од функционалноста на апликацијата?

Табела 18. Резултати за прашање 4 од анкетата
Table 18. Results for question 4 of the survey

Одговори	Број на корисници	Процентуален број на корисници изразени
Не задоволен	0	0,00%
Задоволен	5	17,00%
Многу задоволен	25	83,00%



Графикон 4: Процентуален резултат на прашање 4 од анкетата
Graph 4: Percentage result on question 4 of the survey

5. Дали би продолжиле со користење на апликацијата?

Табела 19. Резултати за прашање 5 од анкетата
Table 19. Results for question 5 of the survey

Одговори	Број на корисници	Процентуален број на корисници изразени
Да	2	7,00%
Не	3	10,00%
Можеби	25	83,00%

Графикон 5. Процентуален резултат на прашање 5 од анкетата
Graph 5. Percentage result on question 5 of the survey

6. Кои се придобивките од ваков тип на апликација?
   - Одговори: Подобра комуникација на далечина, подобра координација во секој момент, поголема флексибилност.
7. Ваше мислење каде најмногу би нашла примена оваа апликација?
   - Одговори: образование, секојдневна комуникација, здравство, состаноци во фирми, презентации.
8. Што дополнително би сакале да вклучува апликацијата?
   - Одговори: подобрување на screen share опцијата.

По сумирање на резултатите од анализата, дојдовме до следните заклучоци:

1. Повеќето анкетирани корисници одговориле дека ваква алатка за синхрона комуникација би ја олесни соработката меѓу луѓето во повеќе области.
2. Ваква апликација повеќе ги привлекува помладите генерации во годишна рамка од 18 до 50, а помалку привлечен за користење за постарите над 50 години.
3. Исто така, голем дел од нив биле задоволни од интерфејсот на овој прототип и едноставноста за управување со истиот.
4. Функционалната на апликацијата е со задоволителна оценка на скалата за задоволство. Некои од нив сметаат дека добро би било да се надгради делот со споделувањето на екран.
5. Корисниците сметаат дека има повеќе придобивки од ваков тип на апликација: подобра комуникација на далечина, подобра координација во секој момент, поголема флексибилност.
6. Многу корисници сметаат дека ваков тип на апликација би бил корисен: во образованието, во здравството, во секојдневната комуникација, за состаноци и презентации.

По сето ова, можеме да заклучиме дека оваа апликација би била од голема корист при каква било соработка во реално време.

Заклучок

Новите технологии секогаш се предизвик и во насока за олеснување и подобрување на работата. Изработката на апликации, особено мобилни апликации кои би ги применувале новите технологии со кои ќе овозможат поефикасен и поефективен начин на работа во секојдневието, секогаш се добредојдени. WebRTC е релативно нова технологија која со своите својства ги задоволува токму тие потреби – олеснување при работа. Наоѓа примени во повеќе области: едукација, здравство, корисничка поддршка, гејминг, со еден збор секаде каде што има било каков вид соработка во реално време.

Со оглед на динамичнитот живот и желбата да бидеме синхронизирани во реално време, WebRTC е технологија која е во паралела тековно актуелна. WEBRTC е токму веб комуникација во реалано време.

Во овој магистерски труд се презентирани карактеристиките на WebRTC, преку имплементација на мобилна софтверска алатка за синхрона соработка т.е. андроид мобилна апликација за видео и аудио повик. Со креирањето на апликацијата и нејзино пошироко користење се утврдени придобивките од WebRTC технологијата, која пред сѐ е бесплатна, а нуди висококвалитетна, со најмала латентност видеокомуникација.

Во овој истражувачки труд за време на евалуацијата е утврдено дека WebRTC како технологија има големи предности и инкорпорирањето на истата во кој било систем од разни сфери е видно ефективна.

Користени кратенки

- WebRTC - Web Real Time Communication
- IoT - Internet of Things
- GIPS - Global IP Solutions
- W3C - World Wide Web Consortium
- IETF - Internet Engineering Task Force
- DTLS - Datagram Transport Layer Security
- SRTP - Secure Real-time Transport Protocol
- HTTPS - Hyper Text Transfer Protocol Secure
- P2P - Peer-to-peer
- API - Application protocol Interface
- ICE - Interactive Connectivity Establishment
- SDP - Session Description Protocol
- SIP - Session Initiation Protocol
- JSON - JavaScript Object Notation
- XHR - XMLHttpRequest
- NAT - Network address translation
- STUN - Session Traversal Utilities for NAT
- TURN - Traversal Using Relays around NAT
- UDP - User Datagram Protocol
- TCP - Transmission Control Protocol
- iSAC - internet Speech Audio Codec
- iLBC - Internet Low Bitrate Codec
- SRTP - Secure Real-Time protocol
- HD - High-definition
- VGA - Video Graphics Array
- QVGA - Quarter Video Graphics Array

Користена литература

1. A. Bergkvist, D. C.Burnett, C. Jennings, A. Narayanan. (2013).WebRTC 1.0: Real-time CommunicationBetween Browsers. Working draft, W3C.
2. A.Johnston, J.Yoakum and K.Singh.(2013). Taking on WebRTC in an enterprise. IEEE Communications Magazine, Vol. 51, No.4.
3. Alan B. Johnston and Daniel C. Burnett.(2014).WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web.
4. Andrii Sergiienko.(2014). WebRTC Blueprints.
5. Burnett, D., T. Brandstetter, C. Jennings, A. Bergkvist, A. Narayanan, and B. Aboba. (2017).WebRTC 1.0: Real-time communication between browsers. W3C working draft, W3C.
6. Cullen Jennings. (2015). WebRTC: Real Time Communications for the Web.
7. Google Developers Codelabs, the free tutorials resources. Real time communication with WebRTC. Преземено на: 7.9.2018 г.https://codelabs.developers.google.com/codelabs/webrtc-web/#0
8. Habibur Rahaman. (2015). A Survey on Real-Time Communication for Web. Scientific Research JournalVolume III, Issue VII.
9. Holmberg, Christer, Stefan Hakansson, and G. Eriksson. (2015). Web real-time communication use cases and requirements. No. RFC 7478.
10. Johnston, Alan, John Yoakum, and Kundan Singh. (2013). Taking on WebRTC in an Enterprise. IEEE Communications Magazine51, no. 4: 48-54.
11. Justin Hart. (2015). WebRTC For Dummies.
12. Joey Lott. (2017). WebRTC: The Ultimate Getting Started Guide.
13. Jansen, Bart & Goodwin, Timothy & Gupta, Varun & Kuipers, Fernando & Zussman, Gil. (2018). Performance Evaluation of WebRTC-based Video Conferencing. ACM SIGMETRICS Performance Evaluation Review. 45. 56-68. 10.1145/3199524.3199534.
14. Koren, István, Petru Nicolaescu, and Ralf Klamma. (2015). Collaborative drawing annotations on web videos. In: International Conference on Web Engineering, 671-674.
15. L. L. Fernandez, M. P. Diaz, R. B. Mejias, F.J. Lopez, J.A. Santos, Kurento. (2013).A media server technologyfor convergent WWW/mobile real-time multimedia communications supporting WebRTC. In: 14thInternational

Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks.

16. M. Handley, C. Perkins, and V. Jacobson. (2006). SDP: session description protocol.

17. M. M. Herterich, C. Peters, F. Uebernickel, W. Brenner and A. A. Neff. (2015). Mobile Work Support for Field Service: A Literature Review and Directions for Future Research. In: 12th International Conference on Wirtschaftsinformatik.

18. Michael Adeyeye, Ishmael Makitla, and Thomas Fogwill.(2015). WebRTC using JSON via XMLHttpRequest and SIPover WebSocket: Initial Signalling Overhead Finding.

19. MDN web docs, free resources for developers. WebRTC API. Преземено на: 7.9.2018 г.https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API

20. Pinikas, Nikos, Spyros Panagiotakis, Despina Athanasaki, and Athanasios Malamos. (2016). Extension of the WebRTC data channel towards remote collaboration and control. In: Proceedings of the International Symposium on Ambient Intelligence and Embedded Systems.

21. Pinikas, Nikos & Panagiotakis, Spyros & Athanasaki, Despina & Malamos, Athanasios. (2016). Extension of the WebRTC Data Channel Towards Remote Collaboration and Control.

22. Pinikas Nikos, Spyros Panagiotakis, Despina Athanasaki, and Athanasios G. Malamos.(2017). A Device Independent Platform for Synchronous Internet of Things Collaboration and Mobile Devices Screen Casting. International Journal of Information and Communication Sciences 2, no. 5.

23. Rob Manson. (2013). Getting Started with WebRTC.

24. RescorlaEric. (2016). WebRTC security architecture.

25. Ryskeldiev, Bektur, Michael Cohen, and Jens Herder.(2018). StreamSpace: Pervasive Mixed Reality Telepresence for Remote Collaboration on Mobile Devices. Journal of Information Processing, 26: 177-185.

26. Salvatore Loreto, Simon Pietro Romano. (2014). Real-Time Communication with WebRTC .

27. Singh, Kundan, and John Buford. (2016). Developing WebRTC-based team apps with a cross-platform mobile framework. In: Consumer Communications & Networking Conference (CCNC), 236-242.

28. SlideShare. WebRTC on mobile. Преземено на: 7.9.2018

r.https://www.slideshare.net/BuraDenizCSM/webrtc-voxxed

29. Trilogy LTE, blog. How WebRTC Revolutionizing Telephony. Преземено на: 7.9.2018 г. http://blogs.trilogy-lte.com/post/77427158750/how-webrtc-is-revolutionizing-telephony

30. Vogt, Christian, Max Jonas Werner, and Thomas C. Schmidt. (2013). Leveraging WebRTC for P2P content distribution in web browsers. In: *Network Protocols (ICNP).*

31. Wikipedia, the free encyclopedia. WebRTC. Преземено на: 7.9.2018 г. https://en.wikipedia.org/wiki/WebRTC

32. WebRTC world, webpage. WebRTC and the call center. Преземено на: 7.9.2018 г. http://www.webrtcworld.com/topics/from-the-experts/articles/368415-webrtc-the-call-center.htm

33. Xing, M., Xiang, S., & Cai, L. (2014). A real-time adaptive algorithm for video streaming over multiple wireless access networks. IEEE Journal on Selected Areas in communications, 32(4), 795-805.

34. Huang, T. Y., Johari, R., McKeown, N., Trunnell, M., & Watson, M. (2015). A buffer-based approach to rate adaptation: Evidence from a large video streaming service. ACM SIGCOMM Computer Communication Review, 44(4), 187-198.

35. Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., & Nießner, M. (2016). Face2face: Real-time face capture and reenactment of rgb videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2387-2395).

36. Ramakrishna, M., & Karunakar, A. K. (2017). SIP and SDP based content adaptation during real-time video streaming in future internets. Multimedia Tools and Applications, 76(20), 21171-21191.

37. Zhang, L., Fu, D., Liu, J., Ngai, E. C. H., & Zhu, W. (2017). On energy-efficient offloading in mobile cloud for real-time video applications. IEEE Transactions on Circuits and Systems for Video Technology, 27(1), 170-181.

38. Koceski, S., & Koceska, N. (2016). Evaluation of an assistive telepresence robot for elderly healthcare. Journal of medical systems, 40(5), 121.

39. Koceski, S., Koceska, N., & Kocev, I. (2012). Design and evaluation of cell

phone pointing interface for robot control. International Journal of Advanced Robotic Systems, 9(4), 135.

40. Loshkovska, S., & Koceski, S. (Eds.). (2015). ICT innovations 2015: Emerging technologies for better living (Vol. 399). Springer.

41. Trajkovik, V., Vlahu-Gjorgievska, E., Koceski, S., & Kulev, I. (2014, September). General assisted living system architecture model. In International Conference on Mobile Networks and Management (pp. 329-343). Springer, Cham.

42. Koceski, S., & Koceska, N. (2011, November). Interaction between players of mobile phone game with augmented reality (AR) interface. In User Science and Engineering (i-USEr), 2011 International Conference on (pp. 245-250). IEEE.

43. Koceski, S., & Koceska, N. (2010, June). Vision-based gesture recognition for human-computer interaction and mobile robot's freight ramp control. In Information Technology Interfaces (ITI), 2010 32nd International Conference on (pp. 289-294). IEEE.

44. Kotevska, O., Vlahu-Gjorgievska, E., Trajkovik, V., & Koceski, S. (2011). Towards a patient-centered collaborative health care system model.

45. Koceski, S., & Koceska, N. (2013). Challenges of videoconferencing distance education-a student perspective. International Journal of Information, Business and Management, 5(2), 274.

46. Petrevska, B., & Koceski, S. (2013). Web-based platform for enhancing tourism development: An exploratory study. Journal of Tourism Challenges and Trends, 6(1), 113.

47. Ananthanarayanan, G., Bahl, P., Bodík, P., Chintalapudi, K., Philipose, M., Ravindranath, L., & Sinha, S. (2017). Real-time video analytics: The killer app for edge computing. computer, 50(10), 58-67.

48. Koceska, N., Koceski, S., Sazdovski, V., & Ciambrone, D. (2017). Robotic Assistant for Elderly Care: Development and Evaluation. International journal of automation technology, 11(3), 425-432.

49. Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H. P., ... & Theobalt, C. (2017). Vnect: Real-time 3d human pose estimation with a single rgb camera. ACM Transactions on Graphics (TOG), 36(4), 44.

50. Amtrup, J. W., Ma, J., Thompson, S. M., Shustorovich, A., Thrasher, C. W., &

Macciola, A. (2017). U.S. Patent Application No. 15/396,306.

51. Kokkonis, G., Psannis, K. E., Roumeliotis, M., & Schonfeld, D. (2017). Real-time wireless multisensory smart surveillance with 3D-HEVC streams for internet-of-things (IoT). The Journal of Supercomputing, 73(3), 1044-1062.

52. Tiberkak, A., Lemlouma, T., Belkhir, A., Bouabdallah, A., & Hentout, A. (2018). A novel approach for generic home emergency management and remote monitoring. Software: Practice and Experience, 48(4), 761-774.