Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Miha Pavlin

Uporaba platforme WebRTC za učinkovitejšo komunikacijo

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Mojca Ciglarič

Ljubljana 2014

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00564 / 2013
Datum: 6.11.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MIHA PAVLIN**

Naslov: **UPORABA PLATFORME WEBRTC ZA UČINKOVITEJŠO KOMUNIKACIJO**
TOWARDS MORE EFFICIENT COMMUNICATION BY USING WEBRTC PLATFORM

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Analizirajte delovne procese in način komuniciranja v izbranem telekomunikacijskem podjetju. Ocenite, kje in kako bi lahko komunikacije izpeljali na bolj učinkovit način in s tem ob zagotavljanju visoke uporabniške prijaznosti zmanjšali stroške. Pozornost usmerite v možnosti uporabe platforme WebRTC, ki omogoča neposredno komunikacijo med dvema brskalnikoma brez nameščanja dodatne programske opreme. Rezultate analize uporabite pri načrtovanju komunikacijske razširitve za spletni brskalnik Chrome. Razširitev implementirajte in ocenite, kako vpliva na znižanje stroškov telefonskih pogovorov v podjetju.

Mentor:
M. Ciglarič
doc. dr. Mojca Ciglarič

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

Dekan:
prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Miha Pavlin, z vpisno številko **63990293**, sem avtor diplomskega dela z naslovom:

Uporaba platforme WebRTC za učinkovitejšo komunikacijo

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Mojce Ciglarič,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 17. februar 2014 Podpis avtorja:

ZAHVALA

Nataši, staršem in podjetju Akton

Kazalo

Seznam uporabljenih kratic

AEC - Acoustic Echo Cancellation
AGC - Automatic Gain Control
AJAX - Asynchronous JavaScript and XML
API - Application Programming Interface
BSD - Berkeley Software Distribution
CDR - Call Data Record
CLID - Calling line identification
CSS - Cascading Style Sheets
CTI - Computer Telephony Integration
DTMF - Dual-tone multi-frequency signaling
HTML - Hyper Text Markup Language
HTTP - HyperText Transfer Protocol
IETF - Internet Engineering Task Force
ICE - Interactive Connectivity Establishment
ISDN - Integrated Services over Digital Network
JSON - JavaScript Object Notation
LCR - Least Cost Routing
LDAP - Lightweight Directory Access Protocol
NAT - Network Address Translation
OAD - Originating Address
PBX - Private Branch Exchange
POP - Point Of Presence
PSTN - Public switched telephone network

RTC - Real Time Communication
RTP - Real-time Transport Protocol
SDH - Synchronous Digital Hierarchy
SDP - Session Description Protocol
SIP - Session Initiation Protocol
SRTP - Secure Real-time Transport Protocol
STUN - Simple Traversak of UDP through NAT
TT - Trouble Ticket
TURN - Traversal Using Relay NAT
UDP - User Datagram Protocol
VOIP - Voice over IP
WEBRTC - Web Real Time Communication

Povzetek

Cilj diplomske naloge je optimizacija delovnih procesov in zmanjšanje stroškov glasovne komunikacije, ki se pojavljajo v telekomunikacijskem podjetju Akton, z uporabo platforme WebRTC. Standard WebRTC nam omogoča glasovno in video komunikacijo med dvema spletnima brskalnikoma brez nameščanja dodatne programske opreme. Platforma WebRTC znotraj spletnega brskalnika vsebuje vse potrebne elemente za zajem, stiskanje, šifriranje in prenos informacij na sprejemno stran. Odjemalca smo realizirali v obliki aplikativne razširitve spletnega brskalnika Chrome za popolno integracijo z obstoječimi spletnimi aplikacijami. Predvsem smo se osredotočili, kako učinkoviteje izrabiti neposredne glasovne povezave z našimi partnerji namesto klasičnega omrežja PSTN nacionalnega operaterja. Rezultat diplomske naloge je razširitev Chrome, s pomočjo katere občutno zmanjšamo stroške telefonskih pogovorov in učinkoviteje izrabimo obstoječe informacije znotraj internih spletnih aplikacij.

Ključne besede: telekomunikacijsko podjetje, WebRTC, Chrome, spletni brskalnik, razširitev, glasovne povezave, stroški, optimizacija

Abstract

This graduate thesis describes solution to optimize efficiency and minimize costs of everyday working process in telecommunication company Akton with use of WebRTC standard. WebRTC enables us to directly communicate between two web-browsers without additional software. It includes all necessary components for capturing, compressing, encrypting and transmitting information to the receiver. User interface was implemented as extension for web-browser Chrome for better integration with existing web applications. Our main focus was on providing best way to reuse existing voice interconnection links with our voice partners instead of classical PSTN network. Result of our work is Chrome extension, which will help us to lower costs of voice communications and better use of information provided by existing web application.

Keywords: telecommunication company, WebRTC, Chrome, web-browser, extension, interconnection links, costs, optimization

Poglavje 1

Uvod

1.1 Predstavitev podjetja

Podjetje Akton je eden izmed večjih alternativnih operaterjev na področju pan-jadranske regije, ustanovljeno leta 1991. Sedež podjetja je v Ljubljani, z manjšimi podružnicami v vseh glavnih mestih bivše skupne države. Primarni dejavnosti podjetja sta ponujanje glasovnih in podatkovnih storitev v veleprodajnem (wholesale) segmentu našim mednarodnim partnerjem. Hrbtenično povezljivost zagotavljata omrežji SDH in Ethernet med točkami POP (Point Of Presence), glavni stični točki s tujimi operaterji sta Frankfurt in Dunaj. Na zasičenem trgu telekomunikacijskih storitev poizkušamo biti vedno korak pred nacionalnimi operaterji na področju odzivnosti, prilagodljivosti pri rešitvah in prilagajanju željam naših strank.

1.2 Motivacija

Velika večina komunikacij s strankami in partnerji poteka preko telefonskih pogovorov z uporabo namiznih telefonov. Vsa dokumentacija, stiki strank in partnerjev, interni imenik, testne številke in zgodovina klicev so v elektronski obliki. Poraja se vprašanje, zakaj pravzaprav ponovno izbirati telefonske številke preko fizične tipkovnice na namiznem aparatu, če jo že imamo v

elektronski obliki.

Drugi, morda v trenutni finančni krizni situaciji še bolj pomemben dejavnik, pa je cena takšnega klica. Akton kot tranzitni operater nima svojega številskega bloka v Sloveniji in glasovnih storitev ne ponuja končnim strankam. Tako za dostop do lokalnega omrežja PSTN uporabljamo storitve drugih operaterjev, kot sta recimo Telekom Slovenije in T2. Tu so cene za vsako minuto pogovora v nacionalne in še posebej v mednarodne destinacije občutno višje kot preko naših glasovnih partnerjev. Že sedaj lahko vsak zaposleni opravi klic tudi preko njih s številsko predizbiro iz namiznega telefona, vendar celoten proces ni uporabniku prijazen. Le malokdo ve, kam je na primer potrebno usmeriti klic za BIH mobile Mostar, da bo za podjetje najceneje. Za vse te težave bo poskrbela naša rešitev. Zmanjševanje stroškov poslovanja je visoko na prioritetni listi vsakega finančno uspešnega podjetja.

Namen diplomskega dela je razviti rešitev, ki bo uporabniku prijazna in hkrati zmanjšala stroške glasovnega komuniciranja. Analiziramo vsakodnevne procese zaposlenih in na podlagi teh dejstev izberemo ustrezno rešitev.

1.3 Vsakodnevni procesi

Z vsakoletno rastjo števila glasovnih povezav z ostalimi operaterji, teh je že več kot 100, podatkovnih vodov in ostalih storitev, se povečuje tudi komunikacija s končnimi strankami in poslovnimi partnerji.

V našem centru za nadzor ekipa 24 ur na dan, 365 dni v letu nadzoruje stanje omrežja in nudi podporo našim strankam ob morebitnih težavah. Prijava težave se začne s poslanim elektronskim sporočilom na naš naslov. Sporočilo avtomatsko prestreže naš sistem za prijavo napak, določi se tip storitve (podatkovna, glasovna) in tako prispe do osebja v podpornem centru. Dežurni na prvem nivoju prejme prijavljeno napako v analizo. Pri podatkovnih storitvah, najetih vodih, se praviloma težave pojavljajo

na zadnjem delu poti, pri tako imenovanem Local Tailu. Seveda so možne težave tudi znotraj hrbteničnega omrežja zaradi prekinjene optike ali slabega vremena v primeru radijskih povezav, vendar so le te redkejše. V večini primerov je treba posredovati prijavljeno težavo naprej ponudniku končnega dela poti in sproti obveščati našo stranko o napredku. Ta komunikacija poteka preko telefonskih pogovorov in elektronske pošte. Osebje uporablja aplikacije:

- sistem za prijavo napak (trouble ticket system, TT)
- nadzor hrbteničnega omrežja (Siemens TNMS)
- stanje na glasovni centrali Teles (iClient),
- kontrolni podatki glasovnih storitev (Captura, CallCheck, pregledovalni zgodovine klicev, periodična poročila po elektronski pošti)
- dokumentacija, seznam partnerjev (Alfresco)
- nadzor strežnikov in ethernet opreme (Zabbix)

Ekipa v nadzornem centru vsakodnevno opravi veliko število testnih klicev preko glasovnih povezav naših partnerjev z namenom, da preveri kakovost ponujenih storitev. Marsikateri poslovni partner se ne drži pogodbeno dogovorjenih minimalnih kakovostnih standardov.

Tehnični oddelek komunicira s poslovnimi partnerji pri aktiviranju novih povezav/storitev oziroma pri reševanju težav z ostalimi tehničnimi ekipami. Uradna naročila morajo biti oddana preko elektronske pošte, druga komunikacija poteka preko telefona zaradi hitrejše odzivnosti. Pri prekinitvah na hrbteničnem omrežju, ko je vzrok pri podizvajalcu storitev, se odpre uradna prijava napake preko pogodbenega elektronskega naslova. Nadaljnje obveščanje poteka preko telefonskih pogovorov, zaključno poročilo RFO (Reason for Outage) se zopet pošlje po elektronskem sporočilu. Tehnični oddelek uporablja vse aplikacije, kot osebje v nadzornem centru, z upravljalnimi pravicami.

Analiza vsakodnevnih procesov znotraj podjetja Akton je pokazala, da so telefonski pogovori najpogostejša oblika komunikacije. Rezultat diplom-

skega dela mora zaposlenim podjetja Akton omogočati preprosto glasovno komunikacijo in hkrati zmanjšati njihove stroške.

Poglavje 2

Izbor primernih tehnologij

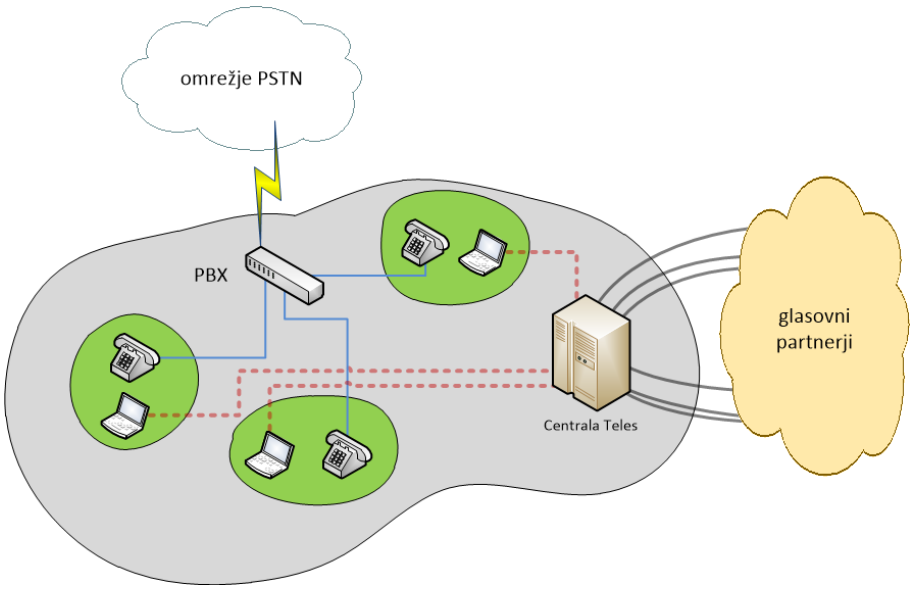
Ideja o povezovanju telefonije z ostalimi namiznimi aplikacijami ni nova, uveljavljen je izraz CTI[8](Computer Telephony Integration). Obstajajo abstraktne rešitve, ki neodvisno od strojne telekomunikacijske opreme ponujajo enotni aplikativni vmesnik za dostop do telekomunikacijskih storitev. Najpomembnejša nabora API na tem področju sta Microsoftov TAPI (Telephony Application Programming Interface) in konkurenčni TSAPI (Telephony Services Application Programming Interface) s strani AT&T. Standard TAPI je bolj usmerjen nadzoru končnih naprav, telefonov, medtem ko TSAPI ponuja boljši nadzor telefonskih central (PBX - Private Branch Exchange). Za te rešitve uporabnik še vedno potrebuje namizni ali programski telefon, ki ga nadzorujemo preko aplikativnih klicev znotraj obstoječih aplikacij, kot je Outlook. V podjetju Akton ne uporabljamo združljive opreme s standardoma TAPI ali TSAPI.

Odločitvi, zakaj za rešitev uporabiti spletno platformo WebRTC znotraj brskalnika Chrome in ne klasične namizne aplikacije (softphone), je botrovalo več dejavnikov:

- Večina aplikacij je spletnih (sistem odpravljanja težav, pregledovalnik zgodovine klicev, spremljanje signalizacije VOIP, dokumentacija znotraj sistema Alfresco) in tako izločimo prenašanje informacij iz

spletnega okolja v namizne aplikacije (kopiraj/prilepi). Na ta način je integracija z obstoječimi aplikacijami popolna, brez spreminjanja slednjih.

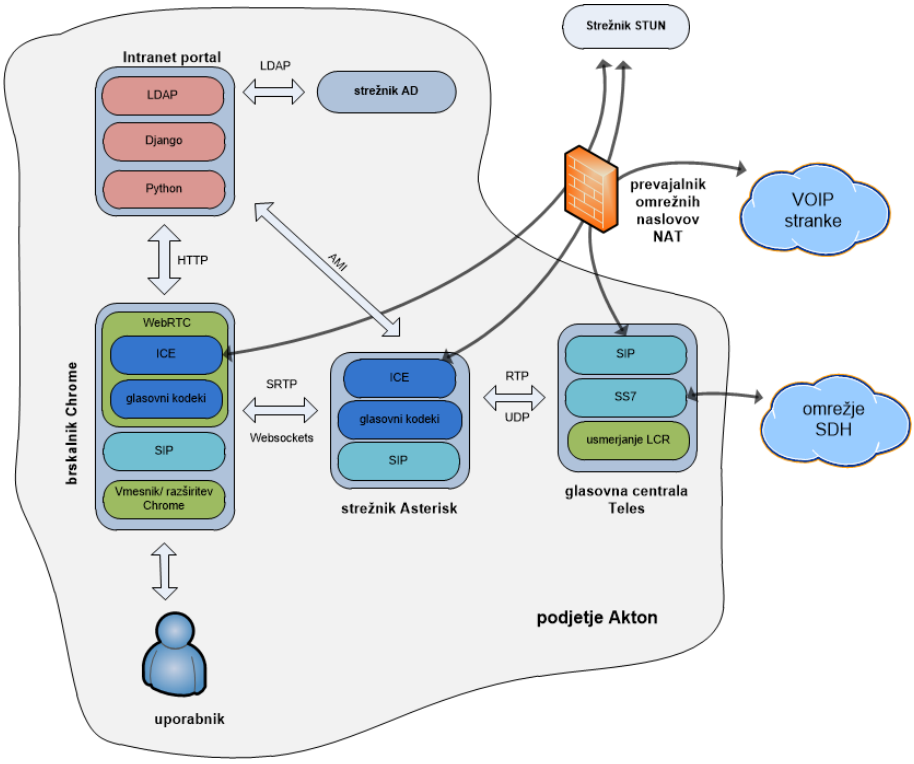
- Uporabljamo operacijski sistem Linux in Windows, pridružujejo se tudi Android tablice, rešimo problem prenosljivosti med platformami, saj brskalnik Chrome deluje na vseh omenjenih operacijskih sistemih. Na uporabnikovem računalniku ni potrebna namestitve dodatne programske opreme.
- Lahka (lightweight) rešitev, brez dodatnih knjižnic, vse potrebno nam ponuja spletni brskalnik Chrome.
- Aktualna tehnologija, ki zna spremeniti način glasovnega komuniciranja. Trenutno zelo popularna je storitev podjetja Skype, ki dobiva resno konkurenco v odprtem standardu WebRTC.



Slika 2.1: Glasovna komunikacija znotraj podjetja Akton.

Slika 2.1 prikazuje trenutni (modre povezave) in zaželeni (rdeče povezave) način komuniciranja v podjetju Akton. Vsako delovno mesto, prikazano z zelenim oblačkom, vsebuje namizni telefon in osebni računalnik. Namizni

telefoni so povezani (modra barva) na hišno centralo (PBX), ki preko zunanjega povezovalnika posreduje klice v omrežje PSTN nacionalnega operaterja. Za potrebe usmerjanje glasovnih pogovorov naših partnerjev uporabljamo telefonsko centralo proizvajalca Teles, ki je namenjena medoperaterskim storitvam. Preko glasovnih povezav na centrali Teles želimo opraviti vse telefonske pogovore zaposlenih, saj so stroški tako občutno nižji kot preko omrežja PSTN. Za primer podamo strošek minutnega pogovora v mobilno omrežje Srbije, ki znaša preko nacionalnega operaterja 13 centov, preko glasovnih povezav naših partnerjev pa 7 centov. Z razvojem odjemalca znotraj spletnega brskalnika Chrome bo možna uporaba rdečih povezav (slika 2.1) in tako prihranek pri stroških glasovnih komunikacij.



Slika 2.2: Odvisnosti uporabljenih tehnologij.

Slika 2.2 prikazuje uporabljene tehnologije in njihovo medsebojno povezavo na strani odjemalca in strežnikov. Uporaba zelo mladega standarda Web-

RTC na strani odjemalca je povečala kompleksnost rešitve na strežniški strani. Neposredna komunikacija med spletnim brskalnikom Chrome in centralo Teles ni mogoča, kajti centrala Teles ne podpira vseh potrebnih tehnologij znotraj standarda WebRTC. Zaradi te omejitve smo uporabili odprtokodno centralo Asterisk, ki skrbi za pretvorbo signalizacijskih in govornih sporočil. Naša razširitev brskalnika Chrome podpira avtentikacijo z domenskim uporabniškim računom, ki se izvede preko internega portala. Slednji je implementiran s pomočjo programskega jezika Python in ogrodja Django. Portal razširitvi brskalnika Chrome posreduje tudi seznam vseh glasovnih medoperaterskih povezav na centrali Teles in telefonski imenik, ki ju prikaže znotraj uporabniškega vmesnika.

2.1 Odjemalec

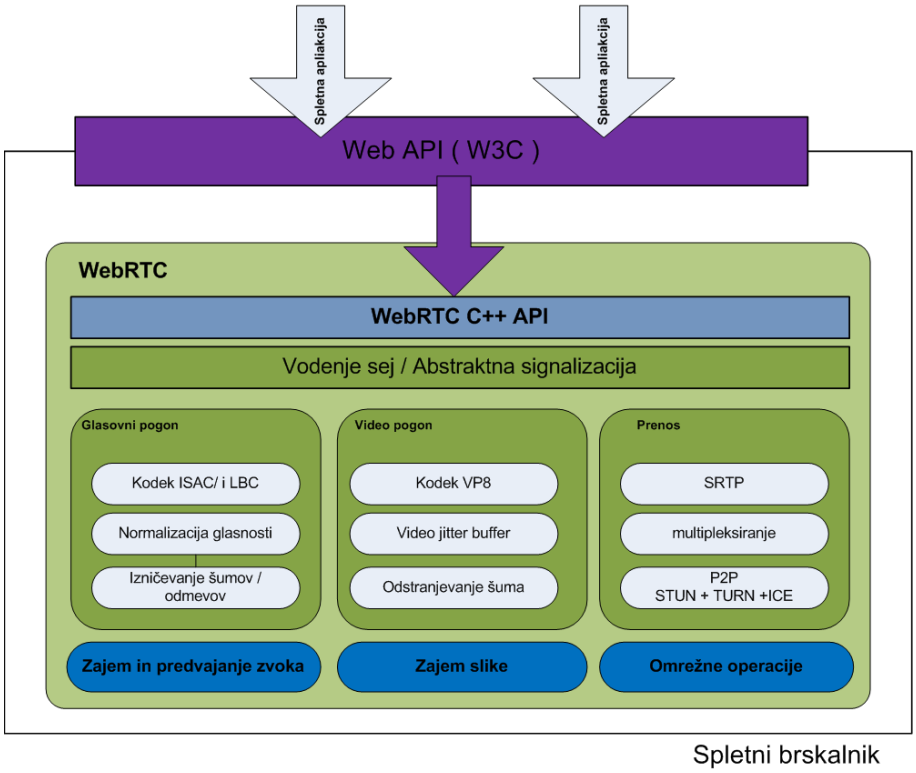
V okviru diplomskega dela vlogo odjemalca predstavlja osebni računalnik z nameščenim spletnim brskalnikom Chrome. Znotraj spletnega brskalnika izkoriščamo platformo WebRTC za zajem in predvajanje zvočnih informacij, pretvorbo, kriptiranje in pošiljanje zvočnih paketov na sprejemno stran. Platforma WebRTC sama po sebi ne vsebuje ustreznega signalizacijskega protokola, zato smo izbrali standardizirani protokol **SIP**, oziroma njegovo implementacijo v obliki odprtokodne Javascript knjižnice **sipML5**. Uporabniški vmesnik smo implementirali v obliki aplikativne razširitve brskalnika Chrome in na ta način omogočili uporabo rešitve na vseh spletnih straneh, ki jih zaposleni podjetja Akton obiskujejo med vsakodnevnimi poslovnimi procesi.

2.1.1 WebRTC

WebRTC[9] je odprtokodni projekt s ciljem omogočiti komunikacijo v realnem času (RTC - Real-Time Communication) med dvema spletnima brskalnikoma. Omogoča video in zvočne prenose ter celo deljenje datotek

brez dodatne programske opreme. V času pisanja standard še vedno ni potrjen, spada pa pod okrilje World Wide Web Consortium (W3C) in IETFa. Platformo je leta 2011 kot odprtokodni projekt predstavil Google, pri razvoju se je nato pridružilo tudi podjetje Mozilla, Microsoft pa je ubral svojo pot s CU-RTC-Web (Customizable, Ubiquitous Real Time Communication).

Slika 2.3 prikazuje arhitekturo platforme WebRTC[10]. S pomočjo aplikativnega vmesnika spletne aplikacije dostopajo do treh medijskih pogonov (zvok, video in podatki). Znotraj vsakega pogona se nahajajo algoritmi za stiskanje/raztegovanje podatkov in izboljšavo kakovosti.



Slika 2.3: Arhitektura platforme WebRTC

Aplikativni vmesnik platforme WebRTC, do katerega dostopamo iz pogona Javascript, je sestavljen iz treh sklopov: **MediaStream** - dostop do vhodno/izhodnih naprav oziroma večpredstavnih tokov, **RTCPeerConnection**

- povezovanje lokalnih in oddaljenih tokov ter **RTCDataChannel** - prenos ostalih podatkov med spletnimi brskalniki.

MediaStream - zajem/predvajanje informacij

Omogoča nam neposredni dostop do komponent na lokalnem računalniku, ki lahko generirajo toke večpredstavnih podatkov (audio, video), imenovani sledi (TRACKS). S klicem funkcije getUserMedia() dobimo instanco objekta MediaStream, ki vsebuje nič ali več sinhroniziranih sledi. Vsak objekt ima izhod in vhod, slednji je odvisen od parametrov pri klicanju konstruktorja. Izsek kode 2.1 prikazuje uporabo objekta MediaStream. Ta kratek primer poveže tok podatkov iz lokalnega mikrofona z **<audio>** HTML5 značko v lokalno zanko, tako da svoj glas slišimo v zvočnikih.

```
function gotStream(stream) {
        // HTML5 <audio> znacka
        output = document.createElement("audio");
        source = window.webkitURL.createObjectURL(stream);
        output.autoplay = true;

        // povezemo <audio> HTML znacko z izvorom/mikrofonom
        output.src = source;
}

navigator.getUserMedia = navigator.getUserMedia || navigator.webkitGetUserMedia;
navigator.getUserMedia( {audio:true}, gotStream );
```

Izsek kode 2.1: Primer uporabe funkcije **getUserMedia()**

Ob vsakem klicu funkcije getUserMedia() se pojavi okno znotraj brskalnika (slika 2.4), ki nas obvesti o zahtevanem dostopu do strojne opreme, v našem primeru mikrofona.

Slika 2.4: Potrditev pravic za dostop

Če se koda izvaja na spletnem mestu HTTPS, do katerega dostopamo preko kriptirane povezave, se izbira shrani. Ob naslednjem klicu getUserMedia() potrditev dostopa do mikrofona ni več potrebna.

RTCPeerConnection - povezovanje tokov

Poslušanje samega sebe kaj kmalu postane dolgočasno, zato se moramo povezati z oddaljenim izvorom medijskega toka, oziroma našim sogovornikom. To dosežemo preko vmesnika RTCPeerConnection. Ker pa se večina naprav/brskalnikov nahaja za usmerjevalniki s tabelami NAT, hitro pride do težav. Če se odjemalec ne zaveda tega dejstva, nastane problem enostranske slišnosti, ko govorni paketi ne dosežejo cilja. Na srečo ima platforma WebRTC integrirano rešitev, ki bolj ali manj uspešno rešuje te težave s pomočjo protokolov **ICE, STUN in TURN**. Ti protokoli so zasnovani za odkrivanje lastnosti preslikovalnih tabel NAT in na podlagi teh podatkov poiščejo optimalno pot za pošiljanje medijskih podatkov med dvema točkama. Platforma WebRTC privzeto uporablja protokol ICE.

- **Protokol STUN[7]** (Simple Traversal of UDP through NATs). Z zahtevkom na zunanji (izven lokalnega omrežja) strežnik STUN dobimo informacijo o našem javnem naslovu IP, načinu preslikave naslova IP znotraj tabele NAT in katera vrata so bila odprta pri pošiljanju zahtevka. Načrtovan je bil z namenom integracije v naprednejše protokole, kot je na primer ICE. Ne deluje ob simetričnih tabelah NAT. Izpis 2.2 prikazuje proženje zahtevka STUN na Googlov javni strežnik.

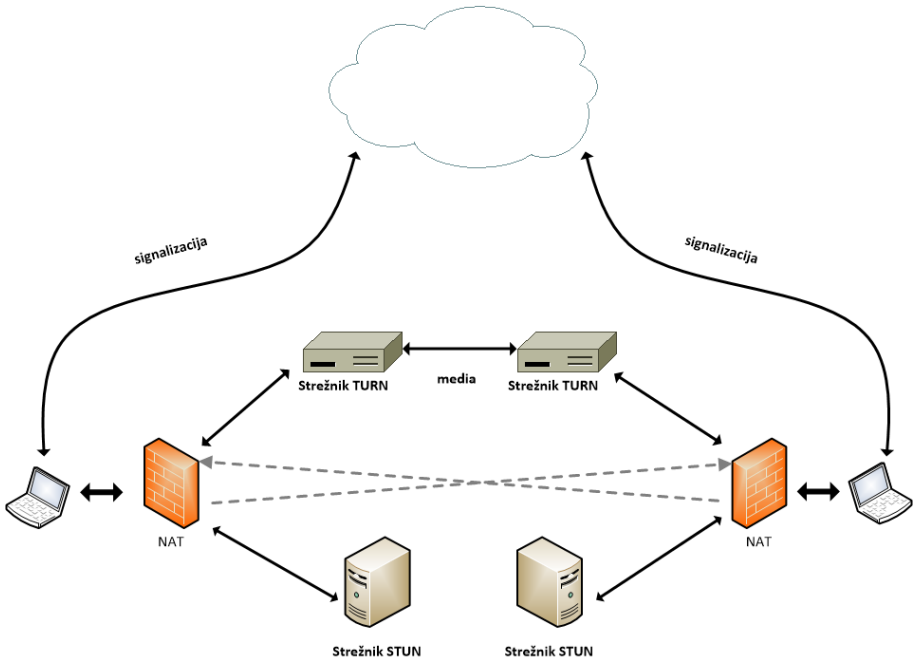
```
$ stun -i eth0 stun.l.google.com:19302
    STUN client version 0.96
    Primary: Independent Mapping, Independent Filter,
                random port, no hairpin
```

Izsek kode 2.2: Uporaba klienta stun znotraj ukazne vrstice

- **Protokol TURN** (Traversal Using Relay NAT) - posredovanje paketov UDP preko zunanjega strežnika. Problem je, ker ne obstajajo javni strežniki TURN zaradi velike potrebne pasovne širine. Poveča se zakasnitev v prenosu. Pojavljajo se tudi varnostni pomisleki pri-

sluškovanja ali spreminjanja vsebine paketov. Optimalno bi bilo, da ponudnik aplikacije WebRTC ponudi svojim uporabnikom javno dostopen strežnik.

- **Protokol ICE[6]** (Interactive Connectivity Establishment) - nadgraditev protokola STUN in TURN z dodatnimi mehanizmi. Slika 2.5 prikazuje diagram poteka iskanja možne poti s protokolom ICE.



Slika 2.5: iskanje poti s protokolom ICE

Potek iskanja možne poti paketov RTP med računalnikoma A in B ob vzpostavitvi seje SIP je naslednji:

- Računalnik A sestavi seznam vseh mrežnih vmesnikov, preko njih sproži zahtevke STUN na javne strežnike.
- Preko vseh mrežnih vmesnikov se izvedejo poizvedbe TURN.
- Prioritizira se pare IP:port po verjetnosti uspešne vidnosti, TURN so najmanj vredni.

- V poslanem sporočilu računalniku B se nahajajo vsi kandidati s prioritetami računalnika A.
- Enak postopek ponovi računalnik B, pridobi listo lokalnih kandidatov in jih posreduje računalniku A.
- Oba računalnika pripravita vse možne kombinacije iz obeh seznamov, duplikati se izločijo.
- Z neposrednimi zahtevami STUN med računalnikoma A in B se preverijo kombinacije. Kjer odgovor doseže izvor, označimo za uspeh.
- Če dosežemo obojestransko povezljivost, sprožimo objavo zvonjenja. Na ta način se izognemo situacijam, ko uporabnik sliši zvonjenje, ob vzpostavitve zveze pa nastopi tišina zaradi blokiranega toka paketkov RTP.

Problematičen je samo scenarij, kjer sta obe strani za simetrično tabelo NAT, kjer nam preostane samo še možnost posredovanje paketkov UDP preko strežnika TURN.

RTCDataChannel - prenos ostalih podatkov

Za prenos ostalih oblik podatkov, kot je na primer medmrežno igranje, deljenje namizja, prenosi datotek, tekstovno sporočanje. Izpis 2.3 prikazuje preprosto aplikacijo za izmenjavo tekstovnih sporočil. RTCDataChannel objekt za delovanje potrebuje delujočo povezavo RTCPeerConnection.

```
var pc = new webkitRTCPeerConnection(ice_servers,{optional: [{RtpDataChannels: true}]});
// posiljanje sporocila
sendChannel = pc.createDataChannel("sendDataChannel", {reliable: false});
sendChannel.send('sporocilo');

// sprejem sporocila
pc.ondatachannel = function(event) {
  receiveChannel = event.channel;
  receiveChannel.onmessage = function(event){
    console.log('Prejeto sporocilo:' +event.data);
  };
};
```

Izsek kode 2.3: Pošiljanje/Prejemanje tekstovnega sporočila

Medijski pogoni

Poleg naštetih programskih vmesnikov so v projekt vključeni tudi trije medijski pogoni: glasovni, video in transportni.

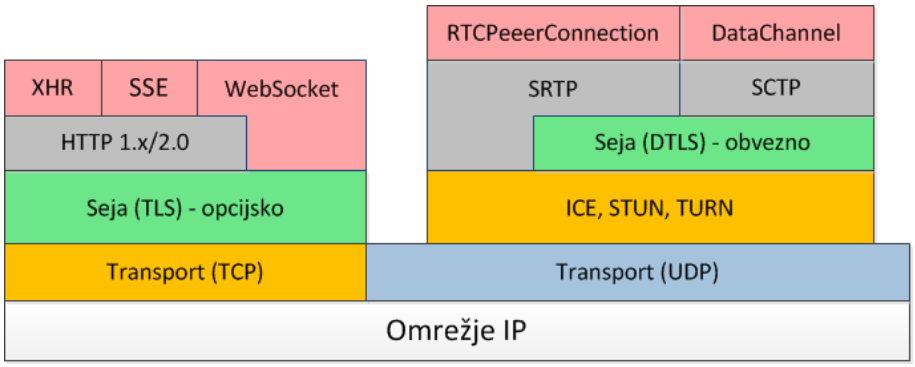
Glasovni pogon (ang. Voice engine) omogoča zajem in predvajanje glasovnih informacij. Kodeki, ki se uporabljajo za stiskanje zvoka, so standardni G.711, iLBC (razvilo ga je podjetje Global IP Sound, širok frekvenčni razpon, odpornost na izgubo paketov, majhna potrebna pasovna širina) in iSAC (robustni, dinamična pasovna širina, potrebuje več pasovne širine kot iBLC). Pogon vsebuje tudi algoritme za izboljšanje kakovosti zvočnih informacij:

- Odstranjevanje odmeva (**Acoustic echo cancellation - AEC**), ki nastane ob različnih situacijah zaradi preobčutljivega mikrofona/preglasnega zvočnika.
- Uravnavanje glasnosti/normalizacija (**automatic gain control - AGC)**
- Odpravljanje ambientalnih šumov (**noise reduction**), kot so zvoki ventilatorja, tipkanje po računalniški tipkovnici, brnenje računalnika.
- Izločanje tišine (**silence suppression**) in s tem preprečevanje prenašanja praznih paketov RTP v omrežju. Daljše obdobje tišine lahko povzroči predčasno zaprtje asociacij v tabeli NAT in s tem izgubo paketov RTP.
- Zaznavanje in generiranje tonov **DTMF**, ki se uporabljajo pri glasovnih odzivnikih (npr. glasovna pošta, navidezne konferenčne sobe)

Video pogon omogoča zajem slike s pomočjo spletne kamere oziroma drugih naprav, dostopnih na lokalnem računalniku. Podatki se stiskajo s pomočjo kodeka VP8 in ne s konkurenčnim H.264 zaradi patentnih omejitev slednjega. Pogon vključuje algoritme za odpravo šuma oziroma zrnatosti slike, ki se pojavi zaradi kombinacije slabih svetlobnih razmer in majhnosti tipal v spletnih kamerah. Prisoten je tudi medpomnilnik za izravnavo trepetanja

(jitter buffer). Z medpomnilnikom se odpravljajo težave zaradi zamud pri paketnem prenosu podatkov.

Standard WebRTC zahteva šifriranje vseh paketnih prenosov. Za prenos medijskih paketov se uporablja nepovezavni protokol UDP, kot je prikazano na sliki 2.6, za kritpiranje teh paketov pa skrbi kombinacija protokolov **DTLS** (Datagram Transport Layer Security) in **SRTP** (Secure Real-Time Transport Protocol).



Slika 2.6: Hierarhija mrežnih protokolov

Sama platforma WebRTC ne vsebuje višjenivojskega protokola za signalizacijo, kot so na primer SIP, H323 in Jingle. Za rešitev v okviru diplomskega dela smo izbrali protokol SIP, implementiran s pomočjo javascript knjižnice, ki se izvaja znotraj spletnega brskalnika.

2.1.2 Signalizacija SIP

Za signalizacijo/vzpostavitev seje med spletnim brskalnikom Chrome in telefonsko centralo Asterisk smo uporabili protokol SIP. V svetu telefonije VOIP se poleg protokola SIP uporablja tudi H.323, vedno bolj prisoten pa je tudi protokol Jingle. Protokol H.323 zaradi kompleksnosti ne obstaja v obliki knjižnice Javascript, ki bi bila primerna za integracijo znotraj spletnega brskalnika. V času nastajanja diplomskega dela nobena implementacija protokola Jingle ni bila funkcionalno primerljiva knjižnicam SIP, zato smo izbrali slednjo.

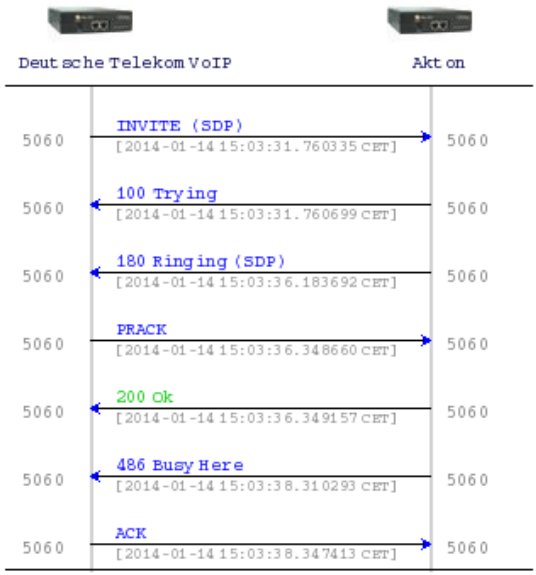
SIP

Signalizacijski protokol SIP sodi med najpomembnejše v svetu telefonije VOIP. Razvoj sta leta 1996 začela Henning Schulzrinne in Mark Handley[13], sedaj pa je pod okriljem Internet Engineering Task Force (IETF). Zadnja standardizirana verzija je opisana v dokumentu RFC 3261. Skrbi izključno za vzpostavitev seje in njeno nadaljnje izvajanje (ang. state machine), ne opredeljuje ničesar na področju prenosa medijskih paketov. Za to poskrbita protokola SDP in RTP. Tipične lastnosti protokola SIP so:

- Neodvisen od transportne plasti, sporočila se lahko prenašajo s pomočjo protokola UDP, TCP ali SCTP.
- Temelji na tekstovnem sporočanju, podobno kot protokol HTTP, s katerim ima nekaj skupnih lastnosti. Recimo vsem znana povratna koda 404 pri obeh protokolih pomeni "Not Found", koda 301 pa "Moved Permanently".
- Zaradi majhnega nabora kontrolnih sporočil in tekstovne narave je relativno preprosto razumljiv. Obstaja pa več kot 50 dodatnih razširitvenih specifikacij RFC in celota postane kompleksna.
- Uporabnike in naprave se identificira preko naslova URI (uniform resource identifier), ki so po formatu podobni elektronskim naslovom (**uporabnik@domena**)

- Privzeta vrata za komunikacijo so 5060
- Sporočila delimo v dve skupini, zahtevki in odgovori
- Zahtevki: REGISTER, INVITE, ACK, CANCEL, BYE, OPTIONS
- Odgovore delimo v dve skupini: **končni (final)** - sem sodijo 2XX, 3XX, 4XX, 5XX, 6XX, vsi ti zahtevajo potrditev prejema **ACK** (acknowledgment) in **začasni (provisional)** - 100 (Trying) ali pa 183 (session progress).

Slika 2.7 prikazuje izmenjavo signalizacijskih sporočil med stranko Deutche Telekom (stran A) in podjetjem Akton (stran B). Stran A sproži zahtevo za klic/vzpostavitev seje v obliki sporočila "INVITE". Stran B odgovori z dvema začasnima odgovoroma (100 Trying in 180 Ringing), ki na strani A sprožita generiranje zvonjenja. Stran A pošlje potrditev v obliki sporočila "PRACK"o prejemu odgovora "180". Stran B pošlje potrditev prejema sporočila "PRACK"v obliki "200 Ok". Kmalu zatem stran B pošlje strani A obvestilo "486 Busy Here", da je klicana številka zasedena. Stran A potrdi sprejeto zavrnitev in klic/seja je končan/-a.



Slika 2.7: Primer izmenjave sporočil SIP

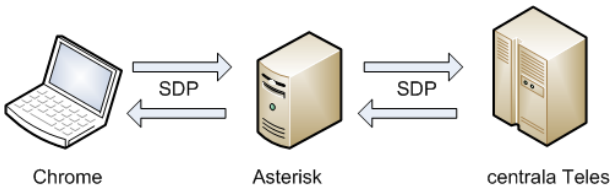
SDP - Session Description Protocol

Protokol SDP skrbi za vzpostavitev medijskega toka podatkov znotraj seje SIP. Temelji na modelu ponudba/odgovor (Offer/Answer). Tako kot protokol SIP uporablja tekstovno obliko sporočil. Protokola samega po sebi ne moremo uporabljati, vključuje se v telo sporočil SIP. Oddajna (stran A) in sprejemna (B) stran s pomočjo protokola SDP določita IP naslove medijskih prehodov (gateway IP address), številke vrat (portov) in kodeka, uporabljenega pri stiskanju podatkov. A stran na podlagi nastavitev medijskega prehoda (media gateway) pripravi ponudbo in jo pošlje strani B. B stran na podlagi predložene ponudbe SDP pripravi odgovor, kjer upošteva svoje parametre/sposobnosti posredovanja medijskih paketkov. V primeru popolnega neujemanja B stran poruši sejo SIP s kodo zavrnitve "488 Not acceptable here". Primer 2.4 prikazuje format sporočila SDP in kratek opis posameznih polj.

```
v=0 //verzija
o=Sonus_UAC 29027 14138 IN IP4 32.58.156.9 //naslov IP
s=SIP Media Capabilities // ime seje
c=IN IP4 32.58.156.9 // naslov IP medijske tocke
t=0 0  // casovne omejitve seje
m=audio 26130 RTP/AVP 0 18 4 2 101 // opis medijske seje: tip,vrata, transportni protokol, seznam kodekov
a=rtpmap:0 PCMU/8000 // lista ponujenih kodekov, 1. izbira G711 ULAW
a=rtpmap:18 G729/8000
a=fmtp:4 bitrate=6.3  //parameteri kodeka
a=rtpmap:101 telephone-event/8000 //DTMF
a=sendrecv
```

Izsek kode 2.4: Primer sporočila SDP

V naši rešitvi se sporočila SDP izmenjujejo med tremi točkami, kot prikazuje slika 2.8.



Slika 2.8: izmenjava sporočil SDP

Knjižnica sipML5 - implementacija protokola SIP

Pri izbiri knjižnice smo morali upoštevati dejstvo, da je v celoti spisana v skriptem jeziku Javascript in tako primerna za izvajanje znotraj spletnega brskalnika. V času nastajanja diplomskega dela so obstajale knjižnice JsSIP, sipML5, Phono in sip-js. Izbrali smo sipML5 zaradi aktivnega razvoja in funkcionalne dovršenosti. V diplomskem delu smo jo uporabili za implementacijo protokola SIP in SDP. Z njeno pomočjo tako vzpostavimo sejo SIP med brskalnikom Chrome in strežnikom Asterisk za izmenjavo kontrolnih sporočil.

sipML5 je odprtokodna knjižnica javascript, izdana pod BSD licenco, v kateri je implementiran popolnoma samostojen klient SIP (UA-user agent). Projekt razvija telekomunikacijsko podjetje Doubango Telecom[3]. Poleg osnovnega zvočnega/video klicanja podpira tudi tekstovno sporočanje, javljanje prisotnosti (presence), zadržanje klica (hold), preusmerjanje in celo deljenje zaslona preko platforme WebRTC. Odlikuje ga preprost API in relativna majhnost na ponujeno funkcionalnost. Za dostop do mikrofona/kamere uporablja WebRTC MediaStream objekt, za povezljivost točk uporablja objekt RTCPeerConnection ali pa WebSockets.
Zaradi preprostega vmesnika API je za osnoven video klic potrebnih samo nekaj vrstic kode, kot je prikazano v primeru 2.5

```
SIPml.init(
    function(e){
        var stack = new SIPml.Stack({realm: 'test.si',impu:'sip:ana@test.si',impi:'ana',password: '*',
            events_listener : { events: 'started', listener : function(e){
                var callSession = stack.newSession('call-audiovideo', {
                    audio_remote: document.getElementById('audio-remote'),
                    video_local : document.getElementById('video-local'),
                    video_remote: document.getElementById('video-remote')
                });
                callSession . call ('joze');
            }
        }
    });
    stack . start ();
    }
);
```

Izsek kode 2.5: Video klic s pomočjo knjižnice sipML5

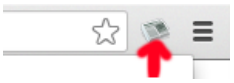
2.1.3 Arhitektura razširitve Chrome

Razširitve[5] (ang. Extensions) so aplikacije, ki razširjajo oziroma spreminjajo funkcionalnost spletnega brskalnika Chrome. Uporabniški vmesnik odjemalca WebRTC smo implementirali v obliki aplikativne razširitve brskalnika Chrome in ne kot klasično spletno stran. Rešitev v obliki razširitve nam omogoča uporabo funkcij platforme WebRTC pri obisku vseh spletnih strani znotraj internega portala in širšega spleta. Pri uporabi klasične spletne rešitve bi morali pred klicanjem izbrane telefonske številke obiskati namensko spletno stran, vnesti številko v spletni obrazec in sprožiti klicanje. Celotni postopek bi bil okoren in ponujal premalo prednosti pred klasično uporabo namiznega telefona.

Ideja razširitev brskalnika Chrome je v tem, da brskalnik privzeto namestimo z osnovnimi funkcijami potrebnimi za uporabo spletnih strani. Napredni uporabniki lahko namestijo različne razširitve, ki na raznorazne načine izboljšajo uporabniško izkušnjo: prikažejo dodatne informacije na spletnih straneh (npr. podatki o slikah, koordinate GPS), omogočajo hitrejšo navigacijo (kretnje z miško, optimizacija zavihkov), prikažejo obvestila itd. Razširitve se nameščajo preko spletnega mesta Google Store ali pa lokalno iz trdega diska.

Razširitve so sestavljene iz množice datotek HTML, prekrivnih slogov CSS, skript JS in slik. V osnovi so to spletne strani z dostopom do večine klicev API, ki jih Chrome ponuja običajnim stranem. Vse datoteke so združene v arhivsko datoteko ZIP in podpisana s certifikatom. Vsaka razširitev ima tako unikatni ID.

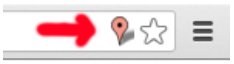
Razširitve lahko komunicirajo s trenutno odprtimi spletnimi stranki, prav tako pa lahko s pomočjo `XMLHttpRequest` pošiljajo/prejemo podatke na/z oddaljenih stežnikov (te pravice moramo eksplicitno dovoliti ob namestitvi). Glede na želeno uporabnost se moramo odločiti med dvema načinoma delovanja.



Slika 2.9: način delovanja - **Browser action**.

V prvem načinu je razširitev aktivna za vsa spletna mesta (browser action), ikona je vedno prisotna v orodni vrstici, kot prikazuje slika 2.9.

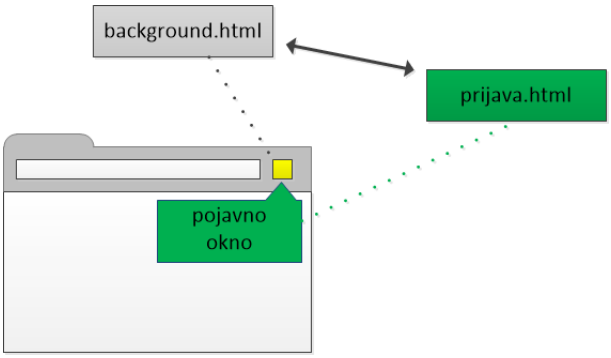
Drugi način delovanja (page action) razširitev aktivira/prikaže samo znotraj naprej določenih spletnih strani, na primer ob obisku naslova www.akton.net. Ikona se prikaže znotraj polja za vnos naslova spletne strani, kot prikazuje slika 2.10.



Slika 2.10: način delovanja - **Page action**.

Naša aplikativna razširitev brskalnika Chrome bo delovala v načinu "Browser action", saj želimo funkcije klicanja ponuditi znotraj vseh spletnih mest.

Razširitev lahko vsebuje uporabniški vmesnik s pojavnim oknom (popup window), ki je implementirano z dokumentom HTML. Rešitev lahko vsebuje več oken, vendar je aktivno/prikazano samo eno. Pojavno okno se prikaže izključno s klikom na ikono razširitve, na sliki 2.11 ikono predstavlja rumeni kvadratek.



Slika 2.11: Arhitektura razširitve Chrome.

Slika 2.11 prikazuje arhitekturo razširitve brskalnika, ki uporablja stran v ozadju (background.html) in eno pojavno okno (prijava.html). Pojavno okno lahko dostopa do funkcij in objekta DOM (Document Object Model) strani v ozadju.

Stran v ozadju - background.html

Večina razširitev brskalnika Chrome ima nevidno, v ozadju izvajajočo stran, imenovano **background.html**. Stran v ozadju vsebuje osrednjo aplikativno logiko brez elementov uporabniškega vmesnika. Pojavna okna (na sliki 2.11 je to prijava.html) pošiljajo asinhrona sporočila strani background.html in ob prejetem odgovoru spremenijo uporabniški vmesnik. Strani v ozadju glede na življenjsko dobo izvajanja delimo v dve skupini:

- Vedno aktivne strani (ang. **persistent background pages**), primerne za dolgo izvajajoče skripte/procese. Treba je paziti na sistemske vire, zato je priporočljivo uporabljati drugo možnost, če okoliščine to dopuščajo.
- Dogodkovne strani (**event pages**), ki se aktivirajo ob zahtevku in ob končanju izvajanja sprostijo pomnilnik in sistemske vire.

Naša rešitev bo uporabljala vedno aktivno stran v ozadju, znotraj katere se bo izvajal sklad SIP v obliki javascript knjižnice sipML5. Na ta način bomo sklad SIP kreirali samo ob aktiviranju razširitve in ne ob vsaki osvežitvi spletne strani.

Razglasna datoteka - Manifest

S pomočjo datoteke **manifest.json** opišemo pomembne lastnosti razširitve brskalnika Chrome z uporabo formata JSON (JavaScript Object Notation). Manifest poveže sestavne dele razširitve v celoto. Nekatere od lastnosti, ki jih definiramo znotraj manifest datoteke:

- Ime razširitve

- Opis razširitve oziroma njen namen
- Seznam vseh pravic, ki jih potrebuje pri delovanju (npr. **audioCapture** za dostop do mikrofona)
- Stran v ozadju: določimo ime skripte in način izvajanja (vedno aktivna ali dogodkovna)
- Seznam elementov, ki tvorijo uporabniški vmesnik: ikona, privzeto pojavno okno

Komunikacija znotraj razširitve

Posamezni deli razširitve brskalnika Chrome seveda potrebujejo medsebojno komunikacijo. Za to poskrbi chrome.extension API, ki vsebuje celoten nabor funkcij za neposredno pošiljanje sporočil med stranmi HTML. S pomočjo funkcije **chrome.extension.getBackgroundPage()** dobimo referenco na javascript window objekt strani v ozadju z aplikativno logiko. Preko nje iz uporabniškega vmesnika dostopamo do spremenljivk in funkcij strani v ozadju, če to seveda potrebujemo. Za dostop v obratni smeri skrbi funkcija **extension.chrome.extension.getViews()**, ki vrača referenco na stran znotraj pojavnega okna.
Dogodki/sporočila se izmenjujemo preko chrome.events API nabora, ki vsebuje funkcijo **chrome.runtime.sendMessage** in registracijo povratne funkcije na dogodek **chrome.runtime.onMessage**. Primer uporabe obeh funkcij prikazuje izsek kode 2.6. Sporočila se izmenjujejo s pomočjo objektov JSON. Vsa komunikacija poteka asinhrono.

```
// V pojavni strani posljemo sporocilo ob pritisku gumba
chrome.runtime.sendMessage({method: "LOGIN"}, function(response) {});

//stran v ozadju prestreze sporocilo
chrome.runtime.onMessage.addListener(
  function(request, sender, sendResponse) {
                if (request.method == "LOGIN") {
                }
  });
```

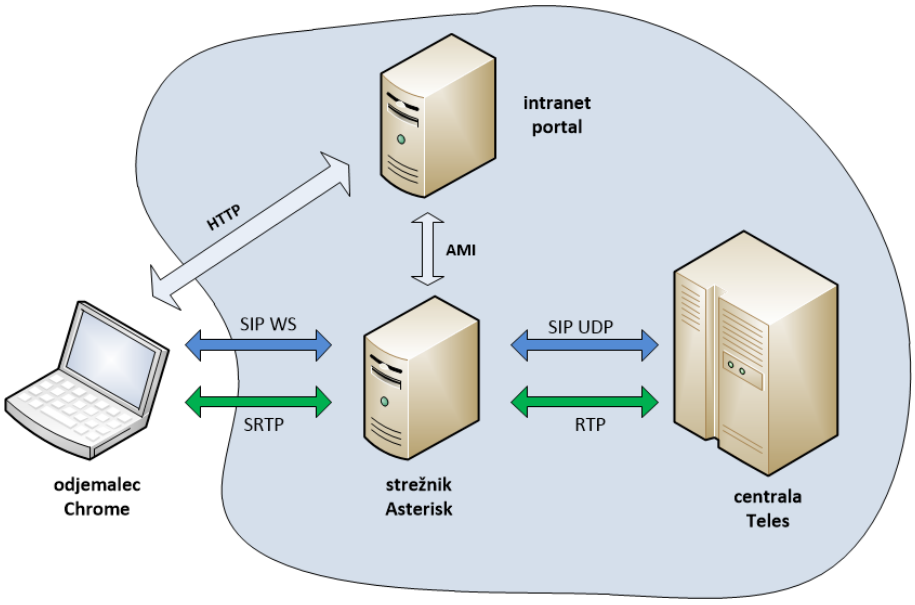
Izsek kode 2.6: Primer izmenjave sporočila.

Sporočila lahko pošiljamo tudi drugim razširitvam, če poznamo njihovo

unikatno številko ID, ki jo dobimo ob podpisovanju datoteke. Ob prijavi povratne funkcije na dogodek **chrome.runtime.onMessageExternal** in definiranju točnega spletnega mesta/domene v datoteki manifest, pa lahko dogodke sprejemamo tudi iz običajnih WEB strani. Obstaja tudi možnost komunikacije z domorodnimi aplikacijami operacijskega sistema. Zaradi številnih možnosti sprejemanja zunanjih sporočil moramo biti pozorni na napade XSS (Cross-site scripting attacks). Vsak odgovor, ki ga sprejmemo iz zunanjih razširitev ali skript, moramo ustrezno preveriti.

2.2 Strežnik

V implementaciji diplomskega dela nastopajo trije strežniški elementi, kot prikazuje slika 2.12.



Slika 2.12: Pregled strežniških elementov.

Intranet portal zagotavlja podporne funkcije za avtentikacijo preko protokola HTTP in nadzoruje delovanje strežnika Asterisk s protokolom AMI. Strežnik Asterisk služi kot signalno-medijski prehod med odjemalcem Chrome in glasovno centralo Teles. Platforma WebRTC za delovanje uporablja protokole,

ki niso prisotne na centrali Teles, zato smo za vmesno točko uporabili strežnik Asterisk. Glasovna centrala Teles skrbi za usmerjanje dohodnega glasovnega prometa preko različnih interkonekcijskih povezav. Klice generirane znotraj brskalnika Chrome želimo zaključiti (terminirati) preko teh povezav.

2.2.1 Asterisk

Asterisk je odprtokodna implementacija telefonske centrale (PBX - Private branch exchange PBX), predstavljena s strani podjetja Digium[2] leta 1999. Vsebuje širok nabor podprtih protokolov, od telefonije IP (SIP, H.323, MGCP), klasičnega ISDN do podpore zgodovinskim analognim telefom. Omogoča neposredno klicanje med končnimi uporabniki, usmerjanje prometa preko javnega omrežja (PSTN) in drugih povezovalnikov (trunks). Sestavljen je iz jedra in razširitvenih modulov.

V verzijo 11 je bila dodana podpora naslednjim protokolom in tako je postal združljiv z ostalimi odjemalci WebRTC:

- SRTP
- WebSockets
- SIP over WebSockets
- ICE
- kodek iSAC (Internet Speech Audio Codec)

V naši rešitvi smo ga uporabili kot signalno-medijski prehod med spletnim brskalnikom in centralo Teles MGC. Neposredno te povezave, brez strežnika Asterisk, ni mogoče izvesti. Brskalnik Chrome pošilja signalizacijske pakete SIP preko WebSockets TCP povezave. Centrala Teles ne podpira standarda WebSockets. Prav tako ni podprt kompresijski kodek ISAC, ki ga uporablja platforma WebRTC znotraj spletnega brskalnika Chrome. Na odhodni strani (proti centrali Teles) se sporočila SIP prenašajo neposredno znotraj sporočil UDP, brez dodatnih aplikacijskih plasti, kot je WebSockets. Kriptirani paketi iSAC SRTP se znotraj strežnika Asterisk pretvorijo (ang. transcoding) v

nekodirane G711 RTP. Tako smo postali združljivi z glasovno centralo Teles, na katero so povezani naši poslovni partnerji.

Razširitev brskalnika Chrome omogoča uporabniku spreminjanje identifikacijske številke (CLID - calling line identification), prikazana klicani osebi ob prejemu dohodnega klica. Implementacijo te storitve smo izvedli s pomočjo protokola AMI (Asterisk Manager Interface), ki omogoča oddaljen nadzor nad delovanjem strežnika Asterisk.

2.2.2 Intranet portal

Aktonov interni portal je implementiran s pomočjo Python Django platforme. Python je priljubljen visokonivojski programski jezik z modularno zasnovo. Z obveznimi zamiki nas prisili v pisanje berljive kode. Odlikuje ga fleksibilna sintaksa, ki nam omogoča reševanje problemov z majhnim številom ukazov. Vsebuje avtomatski nadzor pomnilnika in močno dinamično tipizacijo spremenljivk, podobno kot Javascript.

Django je spletno ogrodje za razvoj aplikacij z modularno zasnovo. Telemlji na zasnovi model-pogled-krmilnik (MVC, model-view-controller). Spisan je v programskem jeziku Python. Zasnovan je z namenom razvoja gradnikov, katere lahko ponovno uporabimo v različnih projektih. S pomočjo samoopažanja (introspekcija) nam pomaga prepoznati podatkovne tipe znotraj objekta/modela in samodejno pripravi programske vmesnike (interface) za dodajanje, brisanje in urejanje modelov znotraj podatkovne baze. Vključuje naslednje komponente:

- Spletni strežnik za pomoč pri razvoju
- Sistem predlog (templates) za hitrejši razvoj spletnih strani
- Validacija in serilizacija spletnih form
- Predpomnjenje (caching) strani

Akton portal vsebuje različne programske module za vsakodnevne procese znotraj telekomunikacijskega podjetja. Nekatere od njih so:

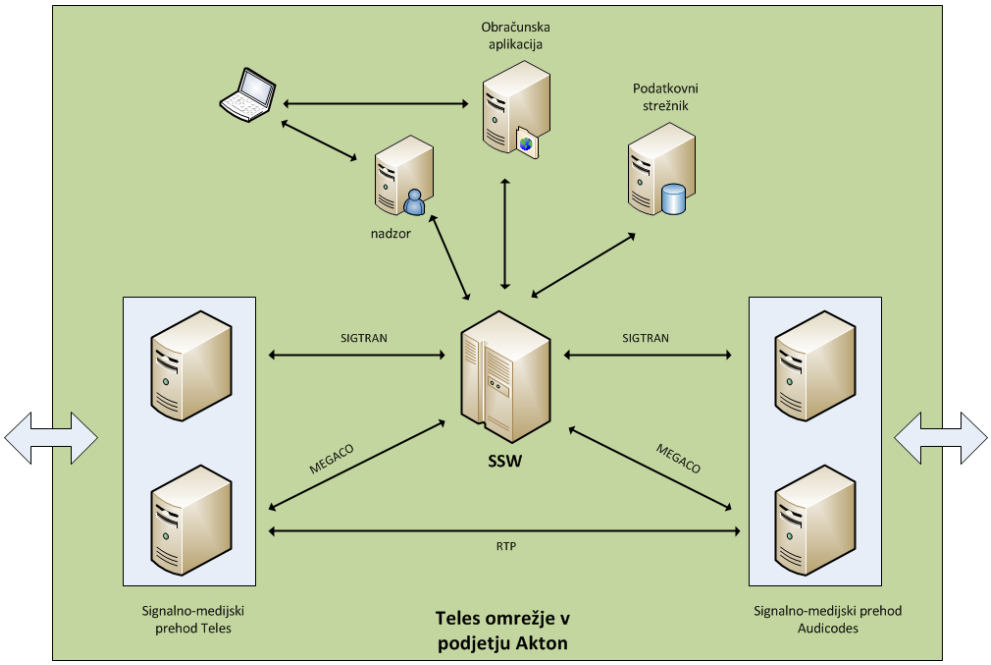
- spremljanje kakovosti in količine glasovnega in podatkovnega prometa za posamezne interkonekcijske povezave
- pregledovalnik zgodovine klicev (CDR - call data record)
- pregledovalnik signalizacije SIP
- dokumentacija podatkovnih vodov
- urejanje dežurstev na prvem in drugem nivoju, beleženje dogodkov med opravljem dežurstva, pregled zgodovine
- prenosljivost številk (MNP - mobile number portability)
- pregledovanje usmerjevalnih tabel na glasovni centrali Teles (routing table)
- imenik stikov

V okviru diplomskega dela smo portal uporabili kot vmesno točko med razširitvijo Chrome, domenskim strežnikom AD (Active Directory) in strežnikom Asterisk. Za avtentikacijo in prijavo na strežnik Asterisk smo želeli uporabiti isti uporabniški račun, kot ga zaposleni uporabljajo pri prijavi na svoj osebni računalnik. Ta uporabniški račun je definiran znotraj strežnika Microsoft Active Directory, ki omogoča dostop preko protokola LDAP (Lightweight Directory Access Protocol). Razširitev brskalnika Chrome s preprostim zahtevkom HTTP pošlje uporabniško ime in geslo portalu, ta posreduje poizvedbo LDAP na strežnik AD, ob prejetem odgovoru poizvedbe LDAP pa portal vrne odgovor JSON brskalniku Chrome. Tako smo se izognili dodatni knjižnici LDAP znotraj brskalnika Chrome (odjemalca). Druga naloga portala je nadzor strežnika Asterisk preko protokola AMI za zagotavljanje prikazovanja želene identifikacijske številke.

2.2.3 Glasovna centrala Teles MGC

Podjetje Akton za usmerjanje glasovnih storitev uporablja centralo Teles MGC, nemškega podjetja Teles z dolgoletno tradicijo. Uvrščamo jo v Class 4[12] razred. V četrti razred kvalificiramo glasovne centrale, ki so namenjene usmerjanju velikega števila tranzitnega/medoperaterskega prometa. Tu so pomembne lastnosti sposobnost posredovanja velikega števila sočasnih klicev

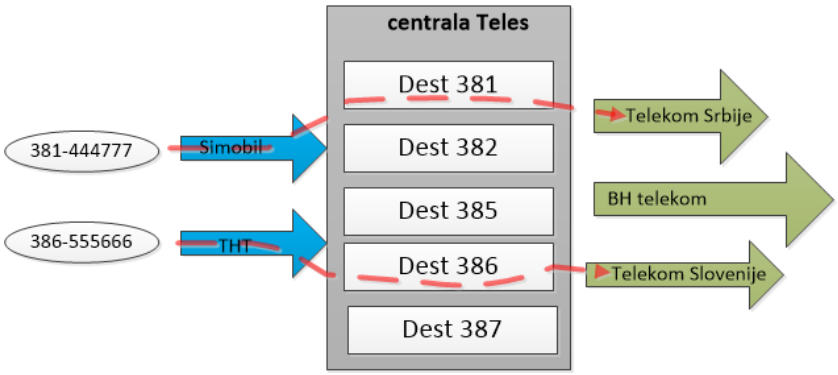
in pretvarjanje med različnimi mediji/protokoli. Praviloma ne podpirajo funkcij namenjenih končnemu uporabniku, ki so domena central Class 5. Temelji na principu omrežja NGN (Next Generation Network), ki uporablja paketni prenos informacij. Osnovna ideja teh omrežij je, da se fizično in logično ločijo transportni elementi (medijski prehodi) od osredje enote, ki skrbi za celotno logiko usmerjanja. Slika 2.13 prikazuje zgradbo omrežja NGN. Starejši modeli telefonskih central so bili zasnovani monolitično.



Slika 2.13: omrežje NGN

Osrednja enota SSW (Soft SWitch) je sestavljena iz dveh strežnikov Sun v sistemu 1+1 (Hot Standby), ki zagotavljata potrebno redundanco. Z omenjeno konfiguracijo lahko dosežemo do 2.000.000 klicnih poizkusov v glavni prometni uri (BHCA - Busy Hour Call Attempts BHCA) in 20.000 vzpostavljenih sočasnih klicev. Za signalizacijske in medijske prehode uporabljamo opremo podjetja Audicodes in Teles. Osrednja enota SSW nadzoruje medijske prehode s pomočjo protokola MEGACO (Media Gateway Control). Usmerjevalne tabele glasovnega prometa se nahajajo v nadzorni

enoti SSW. Usmerjevalno pravilo določa odhodni povezovalnik (trunk) glede na klicano telefonsko številko. Slika 2.14 prikazuje klic na številko 381-444777, ki je bil poslan s strani Simobila. Modre puščice predstavljajo dohodne povezovalnike, zelene pa odhodne. Usmerjevalna aplikacija znotraj centrale je našla pravilo "Dest381". Ta definira, da se vsi klici z mednarodno kodo 381 usmerijo na odhodni povezovalnik Telekoma Srbije.



Slika 2.14: Usmerjanje prometa na centrali Teles MGC.

Usmerjevalne tabele vsakodnevno urejamo preko obračunske aplikacije (billing system). Obračunska aplikacija vsebuje seznam naših poslovnih partnerjev. Vsak partner ima definiran cenik po posameznih mednarodnih kodah. Modul za usmerjanje po najmanjših stroških (LCR - least-cost routing) pripravi optimalna pravila, ki podjetju Akton prinašajo dobiček.

Naša razširitev brskalnika Chrome zahtevani glasovni klic preko strežnika Asterisk pošlje na centralo Teles. Centrala takšen dohodni klic enakovredno ostalim klicem usmerja skladno s pravili. Na takšen način zaposleni znotraj podjetja Akton dostopajo do odhodnih glasovnih povezovalnikov.

Poglavje 3

Implementacija

Implementacijo smo začeli z razvojem strežniških delov rešitve. Znotraj intranet portala smo dodali potrebne funkcije za domensko avtentikacijo preko protokola LDAP. Sledil je strežnik Asterisk. Programsko centralo smo morali zgraditi iz izvorne kode. Spremenili smo ustrezne nastavitve za uspešno povezovanje s centralo Teles na eni strani in brskalnikom Chrome na drugi strani. Na glasovni centrali Teles so bila dodana dohodna povezovalna pravila.

Ob pripravljenem strežniškem delu smo začeli z implementacijo razširitve Chrome, razdeljeno v dve fazi. V prvi fazi smo implementirali stran v ozadju, ki skrbi za dostop do platforme WebRTC, izvajanje sklada SIP in komunikacijo s portalom. V drugi fazi je sledila izgradnja uporabniškega vmesnika ter komunikacija do strani v ozadju.

3.1 Portal in LDAP

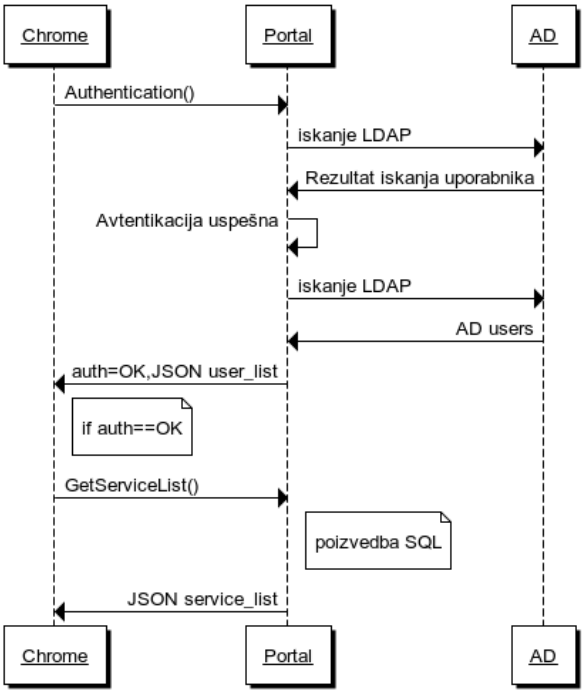
Na spletni strežnik z internim portalom smo namestili knjižnici python-ldap in py-Asterisk. Prva nam omogoča komunikacijo preko protokola LDAP, druga pa oddaljen nadzor strežnika Asterisk. Znotraj platforme Django smo definirali novi razred ActiveDirectory, ki preveri pravilnost posredovanega uporabniškega imena in gesla. Izsek kode 3.1 prikazuje povezovanje na

domenski strežnik Microsoft Active Directory.

```
class ActiveDirectory:
        LDAP_SERVER = 'ldap://192.168.xxx.xxx'
        def check_credentials(self, username, password):
        try:
            self.ldap_client = ldap.initialize(self.LDAP_SERVER)
            self.ldap_client.set_option(ldap.OPT_REFERRALS, 0)
            self.ldap_client.simple_bind_s(LDAP_USERNAME, LDAP_PASSWORD)
            base_dn = 'OU=Uporabniki,DC=akton,DC=loc'
            details = self.ldap_client.search_s(
                                base_dn, ldap.SCOPE_SUBTREE,'mailNickName=' +username,
                                ['cn', 'givenName', 'sn', 'mail', 'mobile','telephoneNumber'])
```

Izsek kode 3.1: Razred ActiveDirectory za preverjanje uporabniškega računa.

Ob pozitivnem (ne praznem) odgovoru na poizvedbo LDAP sprožimo še eno iskanje po vseh domenskih uporabnikih. Tako dobimo seznam vseh sodelavcev z njihovimi mobilnimi in stacionarnimi številkami ter elektronskim naslovom. Vse te informacije vključimo v odgovor JSON, ki ga vrnemo razširitvi Chrome. Celoten potek komunikacije prikazuje slika 3.1.



Slika 3.1: Komunikacija Chrome<->Portal.

Funkcija **getservicelist()** vrača seznam interkonekcijskih partnerjev in njihove usmerjevalne kode na glasovni centrali Teles. Ta seznam urejamo ročno preko spletne strani znotraj portala. Funkcija se pokliče z ločenim zahtevkom AJAX s strani razširitve brskalnika Chrome, odgovor je v formatu JSON. Tako smo se izognili vnašanju seznama direktno v izvorno kodo razširitve, saj se seznam spreminja ob aktiviranju/deaktiviranju glasovnih povezav na centrali Teles.

Razširitev brskalnika Chrome omogoča uporabniku spreminjanje identifikacijske številke (CLID - calling line identification). Funkcija **ChangeOAD** (prikazana v izseku 3.2) z ukazom DBPut preko protokola AMI posreduje identifikacijsko številko strežniku Asterisk, ki jo strani v interno podatkovno bazo imenovano AstDB (Asterisk Database).

```
from Asterisk.Manager import Manager

def ChangeOAD(request):
    pbx = Manager((ASTERISK_IP, 5038), 'akton', '*******')

    family=request.POST.get('family')
    value=request.POST.get('value')

    id = pbx._write_action('DBPut', {'Family': family, 'Key': 'OAD','Val': value})
```

Izsek kode 3.2: Oddaljen nadzor strežnika Asterisk.

3.2 Strežnik Asterisk

Projekt Asterisk je bilo potrebno zgraditi iz izvorne kode, saj najnovejše različice ni na voljo za uporabljeno distribucijo Ubuntu. Postopek gradnje je podrobno opisan na strani razvijalcev in vključuje namestitev vseh odvisnih knjižnic, pripravo datoteke **Make** in na koncu grajenje celotnega paketa. Po končani namestitvi in zagonu strežnika Asterik, smo pravilno delovanje preverili s povezovanjem na upravljalno konzolo z ukazom "asterisk -r". Primer izpisa aktivne upravljalne konzole prikazuje izpis 3.3.

```
$ asterisk -vvvvvvvvvvvr
Asterisk 11.5.0, Copyright (C) 1999 - 2012 Digium, Inc. and others
==========================================
Connected to Asterisk 11.5.0 currently running on Billy (pid = 1503)
Billy*CLI>
```

Izsek kode 3.3: Upravljalna konzola centrale Asterisk.

Namestitvene datoteke se nahajajo v direktoriju **/etc/asterisk**. Potrebno je bilo popraviti privzete nastavitve centrale Asterisk, da smo bili pripravljeni na povezovanje s strani odjemalca Chrome in glasovne centrale Teles. V datoteko **sip.conf** smo dodali uporabniške račune za vse domenske uporabnike, kot uporabniško ime pa smo uporabili stacionarno telefonsko številko zaposlenega. Primer 3.4 prikazuje nastavitve uporabniškega računa za telefonsko številko 38612307268. Ti uporabniški računi se uporabljajo pri povezovanju (registraciji) brskalnika Chrome oziroma knjižnice sipML5 na strežnik Asterisk. V datoteki sip.conf definiramo tudi odhodni povezovalnik do centrale Teles.

```
[38612307268]
nat=no                          //obe tocki sta za tabelo NAT, zato izklopimo to moznost
type=friend                     //omogocimo odhodne in dohodne klice
qualify=no                      //izklopimo peridoicno preverjanje s sporocilom SIP OPTIONS
host=dynamic                    //odjemalci imajo dinamicen naslov IP
context=incoming-internal-SIPP  //zacetna tocka v usmerjevalni tabeli serverja Asterisk
transport=udp,ws,wss            //omogocimo UDP, WebSockets in SecureWebSockets
icesupport = yes                //ICE
videosupport=no                 //ne potrebujemo video povezljivosti
directmedia=no                  //paketki RTP naj gre preko Asteriska, transcoding
disallow=all
allow=ulaw                      //seznam dovoljeni kodekov
allow=alaw
```

Izsek kode 3.4: Nastavitve uporabniškega računa za telefonsko številko 38612307268.

Med strežnikom Asterisk in centralo Teles je uporabljen signalizacijski protokol SIP (slika 2.12). Izsek kode 3.5 prikazuje nastavitve razdelka TelesMGC. Na tej povezavi se ne uporablja avtentikacija, saj se oba strežnika nahajata znotraj lokalnega omrežja, zadostuje nam identifikacija

po naslovu IP.

```
[TelesMGC]
type=friend
fromdomain=akton.net
host=81.17.xxx.xxx                    // definiramo samo staticni IP
insecure=very                         // in izklopimo registracijo
```

Izsek kode 3.5: Lastnosti odhodnega povezovalnika z imenom TelesMGC.

Omogočili smo sprejem sporočil SIP preko protokola WebSockets v datoteki **http.conf**. Strežnik Asterisk sprejema dohodna sporočila WebSockets na vratih 8088. Izpis 3.6 prikazuje vsebino datoteke http.conf.

```
[general]
enabled = yes
bindaddr = 192.168.9.169
bindport = 8088
```

Izsek kode 3.6: Izpis datoteke **http.conf**.

Asterisk za usmerjanje uporablja seznam pravil imenovan **dialplan**, ki se nahaja v datoteki **extensions.conf**. Za nas je pomemben razdelek "**incoming-internal-SIPP**", ki smo ga predhodno navedli pri definiciji uporabnikov v datoteki sip.conf. Razdelek je sestavljen iz petih korakov, prikazanih v izpisu 3.7.

```
[incoming-internal-SIPP]
exten => _X.,n,Set(inTrunk=${CUT(CHANNEL,-,1)})      //1.
exten => _X.,n,Set(inTrunk=${CUT(inTrunk,/,2)})      //2.
exten => _X.,n,Set(OAD=${DB(${inTrunk}/OAD)})        //3.
exten => _X.,n,Set(CALLERID(all)=${OAD})             //4.
exten => _X.,n,dial(SIP/${EXTEN}@TelesMGC)           //5.
```

Izsek kode 3.7: Usmerjevalno pravilo strežnika Asterisk.

S pomočjo prvih dveh ukazov izluščimo niz med znakoma "/"in " -"iz imena dohodnega govornega kanala (channel name), ki predstavlja ime dohodnega povezovalnika. Tretji ukaz izvede poizvedbo znotraj lokalne podatkovne baze AstDB. To vrednost smo predhodno vpisali z zahtevkom DBPut preko protokola AMI. Četrti ukaz nastavi dobljeno vrednost kot identifikacijsko številko, ki se prikaže klicani osebi. Peti korak preko povezovalnik TelesMGC,

ki smo ga predhodno definirali v datoteki sip.conf, sproži odhodni klic.

S tem korakom so nastavitve strežnika Asterisk pripravljene. Izpis 3.8 prikazuje izpis usmerjanja uspešnega klica znotraj administrativne konzole. Uporabnik s številko 38612307268 je opravil klic na številko 38640695504. Klic je bil usmerjen na odhodni povezovalnik TelesMGC z naslovom 81.17.xxx.xxx. Glasovna centrala Teles je za izmenjavo paketov RTP uporabljala vrata 29000, brskalnik Chrome pa vrata 33125. Brskalnik je imel dodeljen naslov IP 192.168.8.200.

```
-- Executing [38640695504@incoming-internal-SIPP:2] Dial("SIP/38612307268", "SIP/38640695504@TelesMGC")
  == Using SIP RTP CoS mark 5
    -- Called SIP/38640695504@TelesMGC
       > 0xa1e3430 -- Probation passed - setting RTP source address to 81.17.xxx.xxx:29000
    -- SIP/TelesMGC-0000001b is making progress passing it to SIP/38612307268
       > 0xa1e3430 -- Probation passed - setting RTP source address to 81.17.xxx.xxx:29000
       > 0xb6b26ad0 -- Probation passed - setting RTP source address to 192.168.8.220:33125
    -- SIP/TelesMGC-0000001b is ringing
    -- SIP/TelesMGC-931-0000001b is making progress passing it to SIP/38612307268
    -- SIP/TelesMGC-931-0000001b answered SIP/38612307268-0000001a
  == Spawn extension (incoming-internal-SIPP, 38640695504, 2) exited non-zero on 'SIP/38612307268'
```

Izsek kode 3.8: Izpis poteka glasovnega klica znotraj administrativne konzole.

S takšnimi nastavitvami smo strežnik Asterisk postavili v vlogo signalno-medijskega prehoda med spletnim brskalnikom Chrome in centralo Teles.

3.3 Glasovna centrala Teles MGC

Naša razširitev brskalnika Chrome ponuja dva načina klicanja izbrane telefonske številke, preko poljubne glasovne interkonekcijske povezave oziroma v načinu najcenejšega usmerjanja (LCR). Za uporabo usmerjanja LCR ni bilo potrebno dodajati oziroma spreminjati usmerjevalnih pravil. Ta pravila se vsakodnevno osvežujejo s pomočjo usmerjevalnega modula znotraj obračunske aplikacije. Za usmerjanje našega klica iz razširitve brskalnika Chrome se tako uporabljajo enaka pravila, kot za ostali dohodni promet naših poslovnih partnerjev.

Usmerjevalna tabela znotraj glasovne centrale Teles je sestavljena iz več tisoč pravil. Vsakemu dohodnemu klicu se na osnovi klicane številke dodeli

ustrezno odhodno usmerjevalno pravilo. Primer takega pravila prikazuje izpis 3.9. Pravilo je sestavljeno iz bloka **match()**, ki definira potrebne pogoje za dodelitev pravila. V našem primeru je to klicana številka, ki se mora začeti z nizom 1784. Sledijo bloki **share()**, ki predstavljajo izbrane odhodne povezovalnike. V primeru 3.9 imamo definirane tri odhodne povezovalnike z imeni to_oper_1, to_oper_2 in to_oper_3 ter blok **reject()** za zavrnitev klica, ki se uporabi v primeru neuspešnega povezovanja.

```
# LCR-ROUTING 01. PREMIUM routing
# DESTINATION St. Vincent
route(Dest_3071){
   match(){
      dad="^..1784.*";              // klicana  stevilka  vsebuje kodo drzave 1784
   }
   share(to_oper_1);                // 1.  izbira
   share(to_oper_2);                // 2.  izbira
   share(to_oper_3);                // 3.  izbira
   reject(){cau="CAU_NOCAV";}       // zavrnitev  klica
}
```

Izsek kode 3.9: Prelivno usmerjevalno pravilo.

Prikazano usmerjevalno pravilo uporablja princip prelivnega usmerjanja (ang. overflow routing), saj vsebuje več kot en blok share(). Večina pravil je sestavljena iz več možnosti na odhodni strani za primere, ko je bil poizkus preko prvega operaterja/povezovalnika neuspešen. Če je klic preko izbranega povezovalnika zavrnjen z ustrezno kodo zavrnitve (Release Cause Code), glasovna centrala Teles klic preusmeri preko naslednjega povezovalnika, ki ga definiramo znotraj usmerjevalnega pravila. Zavrnitvena koda 17/USER BUSY ne sproži prelivnega mehanizma, saj je klicani uporabnik že v aktivnem telefonskem pogovoru. Prelivna pravila nam pri rešitvi znotraj diplomskega dela zagotavljajo visoko stopnjo uspešnih klicnih poizkusov, saj nam glasovna centrala Teles avtomatično preusmerja odhodne klice na delujoče povezovalnike.

Razširitev brskalnika Chrome omogoča klicanje tudi preko točno določenega odhodnega povezovalnika. V tem primeru ne smemo uporabiti prelivnega usmerjanja. Tak način klicanja se uporablja v primerih, ko moramo testni klic opraviti preko glasovne interkonekcijske povezave, za katero smo dobili

prijavo napake. Tak način usmerjanja smo realizirali s številsko predizbiro. Za vsak odhodni povezovalnik, interkonekcijsko povezavo, imamo rezervirano tri mestno kodo. Ta številka se nahaja pred klicano številko. Primer 3.10 prikazuje usmerjevalno pravilo za klice preko Deutche Telekoma.

```
/**
 * Service 0.2.1
 *
 * To: Deutsche Telekom FC
 */
route( service_0_2_1_asterisk_route ){
  match() {
    pc = "asterisk";
    dad = "^..021.*";
  }
  subst(){
    dad = "s/\(^..021\)\(.*\)/\ii\2/";
    dno="r/E522/";
  }
}
```

Izsek kode 3.10: Usmerjevalno pravilo s številsko predizbiro.

Deutche Telekom ima dodeljeno kodo 021. Za testni klic na številko 38641555444 preko Deutche Telekoma moramo tako izbrati številko 02138641555444. Primer 3.10 ima v bloku match() navedena dva pogoja. Prvi je dohodni povezovalnik, klic mora prispeti na glasovno centralo s strani strežnika Asterisk. Drugi pogoj je klicana številka, ki se mora začeti z nizom 021. Klic, ki zadostuje obema pogojema, usmerimo na odhodni primarni port E1 z indexom 522 in popravimo klicano številko (odrežemo niz 021) za pravilno usmerjanje pri končnem operaterju.

3.4 Razširitev brskalnika Chrome

Razširitev uporablja standardne elemente HTML, slogovni jezik CSS in tri zunanje knjižnice. Poleg SipML5, ki definira sklad SIP, sta tu še jQuery in jQueryUI. Prva nam olajša iskanje elementov znotraj drevesa DOM, ki sestavlja dokument HTML, jQueryUI pa razširi standardne elemente HTML z lepšim videzom in dodatnimi funkcijami.

Izsek kode 3.11 prikazuje razglasno datoteko manifest.json, ki opredeljuje

lastnosti razširitve Chrome.

```
{
   "name": "Akton DIALER",
   "background": {
      "persistent":true,
      "scripts": ["SIPml-api.js","background.js"]
   },
   "browser_action": {
      "default_icon": "icon.png",
      "default_popup": "login.html"
   },
   "permissions":["tabs","contextMenus","notifications","storage","audioCapture","webRequest"],
}
```

Izsek kode 3.11: Razglasna datoteka **manifest.json**.

Razširitvi brskalnika Chrome smo dodelili ime "Akton DIALER"preko lastnosti **name**. Stran v ozadju (background.html) se izvaja v načinu **persistent**, ki je oznaka za vedno aktivno stran v ozadju. Na ta način je sklad SIP vedno aktiven, registriran na strežnik Asterisk in posluša na dohodna sporočila. Polje **scripts** vsebuje seznam skript Javascript, ki tvorijo stran v ozadju. Akton Dialer uporablja dve. Sklad SIP znotraj knjižnice sipML5 se nahaja v datoteki SIPml-api.js, background.js pa vsebuje vso ostalo aplikativno logiko za komunikacijo z uporabniškim vmesnikom in intranet portalom.

Lastnost **default_icon** vsebuje ime grafične datoteke z ikono, ki se prikaže v orodni vrstici spletnega brskalnika Chrome. Slika 3.2 prikazuje ikono razširitve Akton DIALER.

Slika 3.2: Ikona razširitve

Polje **default_popup** definira privzeto pojavno okno. V primeru razširitve Akton DIALER je to prijavna forma za vnos uporabniškega imena in gesla, nahaja se v datoteki login.html. Potrebne pravice za delovanje razširitve brskalnika Chrome so naštete v polju **permissions**:

- **tabs**: dostop do odprtih zavihkov
- **contextMenus**: možnost desnega klika
- **notifications**: obvestila, status klica
- **storage**: shranjevanje nastavitev preko chrome.storage APIja
- **audioCapture**: dostop do mikrofona
- **webRequest**: dovolimo zahtevke AJAX do intranet portala

Stran v ozadju - background.js

S pomočjo dogodka **window.onload**, ki se izvede ob popolni naložitvi spletne strani, kreiramo značko HTML5 **<audio>**. Primer 3.12 prikazuje kreiranje značke, preko katere predvajamo dohodni govorni kanal.

```
window.onload = function() {
    audioRemote = document.createElement("audio");
    audioRemote.autoplay=true;
    document.body.appendChild(audioRemote);
}
```

Izsek kode 3.12: Dogodek window.onload

Definiramo povratni funkciji za sprejem signalizacijskih sporočil SIP in ju kot parametra podamo konstruktorju objekta **SIPml.Stack()**. Izsek kode 3.13 prikazuje funkciji onSipEventStack in onSipEventSession.

```
    // Callback function for SIP Stacks
    function onSipEventStack(e) {
        switch (e.type) {
            case 'started':
            break;
            case 'stopped':
            break;
            case ....
        }
    }
    // Callback function for SIP sessions (INVITE, REGISTER, MESSAGE...)
    function onSipEventSession(e) {
        switch (e.type) {
            case 'connecting': case 'connected':
            break;
            case ...
        }
    }
```

Izsek kode 3.13: Sprejem sporočil SIP.

Povratna funkcija za sprejem sporočil iz pojavnega okna/uporabniškega vmesnika. Implementirane so funkcije za prijavo in odjavo preko intranet portala ter vzpostavitev in rušenje klica. Koda 3.14 prikazuje registracijo anonimne funkcije na dogodek **onMessage**.

```
chrome.runtime.onMessage.addListener(function(message,sender,sendResponse){

    if (message.method=='ADlogin') {
        LoginToAD(message.txtUsername,message.txtPassword);
    } else
    if (message.method=='sipCall') {
        sipCall(message.number);
    } else
    if (message.method=='LogOut') {
        sipUnRegister();
    }
    if (message.method=='HangUp') {
        sipHangUp();
    }

});
```

Izsek kode 3.14: Sprejem sporočil uporabniškega vmesnika.

V razširitvi Akton DIALER za prijavo uporabimo domenski uporabniški račun. Preko metode AJAX POST se pošljeta uporabniško ime in geslo na intranet portal. Portal z zahtevkom LDAP na strežnik Microsoft Active Directory preveri pravilnost uporabniškega računa. Odgovor na zahtevek Ajax POST ob uspešni prijavi vsebuje stacionarno in mobilno številko uporabnika. Stacionarna številka se v nadaljevanju procesa uporabi pri registraciji na strežnik Asterisk kot uporabniško ime.

Ob pravilno vneseni kombinaciji računa in gesla je v odgovoru JSON vrednost polja "authorized"pozitivna in nadaljujemo z inicializacijo knjižnice SIPml. Objekt SIPml sproži registracijo SIP na strežnik Asterisk. Ker smo izbrali dolgo izvajajočo stran v ozadju, ostanemo prijavljeni na strežniku Asterisk do ročne odjave oziroma izhoda iz spletnega brskalnika. Funkcija, zadolžena za preverjanje uporabniškega računa, se imenuje LoginToAD() in je prikazana v izseku kode 3.15.

```
function LoginToAD(username,password)
{
    options = {};
    $.ajax({
        url: 'https://billy2.akton.net/Portal/webrtc_login/',
        type: (options.type || 'POST'),
        data: { txtUsername: username, txtPassword: password },
        success: function(data, textStatus, jqXHR) {
            if(typeof data == "object") { //response is assumed to be JSON
                if (data.authorized==true) {
                    num = data.number;
                    SIPml.init(postInit);
                    sipRegister("sip:"+num+"@akton.net",null);

                } else {
                    RemExecute({method:"InvalidADLogin"});
                }
            }
        }
    });
}
```

Izsek kode 3.15: Zahtevek AJAX za preverjanje uporabniškega računa.

Ob neuspešni prijavi se preskoči inicializacija sklada SIP in obvestimo uporabnika o napaki. Ker stran v ozadju ne prikazuje uporabniškega vmesnika, pošljemo sporočilo o napaki prijavnemu oknu znotraj skripte **login.js**. To je realizirano s klicem funkcije **RemExecute**, prikazana v izseku 3.16.

```
function RemExecute(msg) {
    chrome.runtime.sendMessage(msg);
}
```

Izsek kode 3.16: Funkcija za pošiljanje sporočil.

Funkcija za registracijo sprejme dva parametra, vendar se geslo trenutno ne uporablja pri registraciji. Preden sprožimo registracijo, initializiramo sklad SIP preko klica konstruktorja **SIPml.Stack()**. Postopek prikazuje izsek kode 3.17.

```
function sipRegister(txtPublicIdentity,txtPassword) {
        oSipStack = new SIPml.Stack({
                        realm: 'akton.net',                    //Domena
                        impu: txtPublicIdentity,      //nas javni naslov v formatu URI sip:<stevilka>@akton.net
                        password: txtPassword,        // geslo, prazno
                        websocket_proxy_url: 'ws://192.168.9.169:8088/ws',  //naslov streznika Asterisk, WS
                        events_listener : {events:'*', listener : onSipEventStack }, // povratna funkcija za dogodke
                }
```

```
        );
        if (oSipStack.start() != 0) {   //zazenemo sklad SIP, ki opravi registracijo
                ShowNotif('Failed to start the SIP stack');
        } else return;
}
```

Izsek kode 3.17: Zagon sklada SIP.

Pojavni okni - uporabniški vmesnik

Uporabniški vmesnik razširitve Akton DIALER tvorita dve pojavni okni. Privzeto, ko uporabnik ni registriran na strežnik Asterisk, se prikaže stran za prijavo (slika 3.3). Stran za prijavo je definirana znotraj datotek **login.js** in **login.html**.

Slika 3.3: Prijavno okno

Prijavna stran vsebuje preproste funkcije za preverjanje vnosnih polj, prikaz obvestila ob neuspešni prijavi in podporne metode za komunikacijo s stranjo v ozadju. S pomočjo HTML5 elementa **localStorage** shranimo zadnje vneseno uporabniško ime za hitrejšo prijavo. Zaradi varnostnega mehanizma znotraj aplikativnih razširitev brskalnika Chrome (Content Security Policy - CSP), se pri grajenju pojavnega okna srečamo z omejitvijo v obliki nedelovanja medvrstičnega (inline) dodeljevanja funkcije dogodku. Znotraj običajne strani HTML lahko elementu tipa gumb (button) dodelimo dogodek **onclick** na naslednji način: **<input type="button"onclick="dogodek()"/>**. Medvrstično dodeljevanje funkcij znotraj razširitev brskalnika ne deluje in tako se klic funkcije dogodek() ob kliku gumba ne bo izvedel. Rešitev za to omejitev je podana v izseku kode 3.18.

```
$(document).ready(function() {
        loadCredentials();
        $('#login').onclick = logIn;
});
```

Izsek kode 3.18: Dodeljevanje funkcije dogodkom.

S pomočjo knjižnice jQuery in operatorja **$** pridobimo referenco do elementa gumb z oznako (lastnost ID) "login". Preko reference na element lahko dodelimo dogodku "onclick"funkcijo logIn().

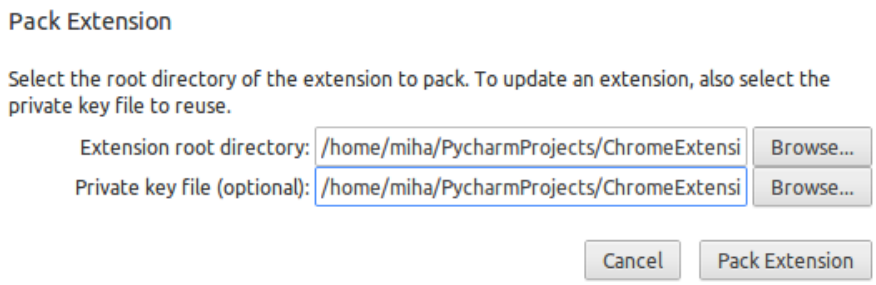
Drugo pojavno okno se pojavi ob uspešni registraciji na strežnik Asterisk in predstavlja osrednji uporabniški vmesnik razširitve Akton DIALER. Pojavno okno je sestavljeno iz dveh datotek dialer.html in dialer.js. Ob proženju dogodka window.onload s pomočjo funkcije getBackgroundPage() pridobimo referenco do strani v ozadju z aplikativno logiko (izsek 3.19). Znotraj strani v ozadju se nahaja objekt JSON **xcentric_list** s seznamom glasovnih partnerjev na centrali Teles MGC.

```
window.onload = function() {
    var xcentric_list = chrome.extension.getBackgroundPage().xcentric_list;

    FillServiceList ( xcentric_list . services );
};
```

Izsek kode 3.19: Dostop do strani v ozadju.

Po končanem razvoju je bilo treba razširitev stisniti in podpisati, da je bila primerna za namestitev ostalim uporabnikom. Brskalnik Chrome ima v razvojnem načinu (developer mode) dodan obrazec za stiskanje, prikazan na sliki 3.4. Podali smo korensko mapo razširitve in opcijsko datoteko PEM z javnim in zasebnim ključem. Zgoščena vrednost javnega ključa se uporabi za unikatno številko ID podpisane razširitve brskalnika Chrome. Po končanem stiskanju smo imeli datoteko s končnico CRX, ki je bila primerna za namestitev zaposlenim podjetja Akton.



Slika 3.4: Stiskanje in podpisovanje razširitve.

3.5 Problemi pri implementaciji

Odjemalca WebRTC smo začeli razvijati kot običajno spletno stran znotraj internega portala. Razvoj razširitve brskalnika Chrome je namreč zaradi omejenih postopkov razhroščevanja kode Javascript veliko počasnejši v primerjavi s klasičnim spletnim programiranjem. Ko smo imeli delujočega odjemalca znotraj platforme WebRTC, smo se lotili migracije v okolje brskalnika Chrome. Zaradi podobnosti s klasičnimi spletnimi stranmi je bil napredek zelo hiter, vendar se je kmalu pojavila ogromna prepreka. Ob navidezno delujoči razširitvi brskalnika Chrome nam ni uspelo sprožiti odhodnega glasovnega klica. V konzoli razhroščevalnika Javascript smo kmalu odkrili problem v sporočilu, prikazanem v izpisu 3.20.

```
getUserMedia error:
NavigatorUserMediaError {constraintName: "", message: "", name: "PERMISSION_DENIED"}
```

Izsek kode 3.20: Napaka pri dostopu do mikrofona.

Vzrok težave je dostop do objekta MediaStream preko klica funkcije **getUserMedia()** znotraj razširitve brskalnika Chrome. Do verzije 28 so razširitve brez uporabnikove vednosti dostopale do mikrofona/kamere, kar je seveda nezaželeno obnašanje aplikacije iz stališča varnosti in zasebnosti. Hitro je sledil popravek, ki je popolnoma onemogočil uporabo objekta Mediastream.

Ekipa razvijalcev se problema zaveda in za prihodnost obljubila vrnitev dostopa do naprav preko klica **getUserMedia**, ne obstaja pa še konkretna rešitev. Uporabniku mora biti v času uporabe mikrofona to jasno prikazano. Ali je to ikona v naslovni vrstici, indikator znotraj vrstice opravil operacijskega sistema ali pa kaj tretjega, še ni znano.

Na našo srečo so se s to omejitvijo morali spopasti[4] tudi razvijalci brskalnika Chrome. Navidezna tipkovnica znotraj operacijskega sistema Chrome OS je implementirana v obliki razširitve. Dodano ima tudi možnost prepoznave govora in s tem potrebuje dostop do mikrofona. Razvijalci so preprosto dodali navidezno tipkovnico na belo listo (white list) razširitev, ki imajo polni dostop do naprav, znotraj izvorne kode spletnega brskalnika. Omenjeno rešitev smo uporabili tudi za našo razširitev Akton DIALER. Seveda je to pomenilo prevajanje izvorne kode brskalnika Chrome.

3.6 Gradnja brskalnika Chrome

Postopek gradnje[1] izvorne kode je na srečo dobro opisan s strani razvijalcev in ni povzročal posebnih težav. Uporabljena so eksotična interna orodja podjetja Google:

- **gclient**: Ovojna skripta za prenos izvorne kode iz repozitorija SVN.
- **gyp**: Python orodje za generiranje platformsko neodvisnih projektov (Makefile, Visual Studio in Xcode), odvisnostna pravila.
- **ninja**: Orodje za gradnjo, podobno kot **GNU Make**.

Po uspešnem večurnem grajenju smo imeli delujoč brskalnik, vendar še vedno brez popravkov, potrebnih za pravilno delovanje razširitve Akton DIALER. Po navodilih[4] smo dodali v datoteki **chrome/browser/media/media_capture_devices_dispatcher.cc** unikatno število ID razširitve Akton DIALER na beli seznam, ki imajo neomejeni dostop do mikrofona. Funkcijo z belo listo **IsMediaRequestWhitelistedForExtension()** prikazuje izsek kode 3.21.

```
// This is a short-term solution to grant microphone access to virtual keyboard
// extension for voice input. Once http://crbug.com/292856 is fixed, remove this
// whitelist.
bool IsMediaRequestWhitelistedForExtension(
    const extensions::Extension* extension) {
    return (extension->id() == "mppnpdlheglhdfmldimlhpnegondlapf") ||
           (extension->id() == "meknneaddcmjndlnngcofmggjemdbjbg") ||
           (extension->id() == "fibgoopalkakpgippfbbbfckmijcbcmk");          //nas dodatek
}
```

Izsek kode 3.21: Bela lista razširitev.

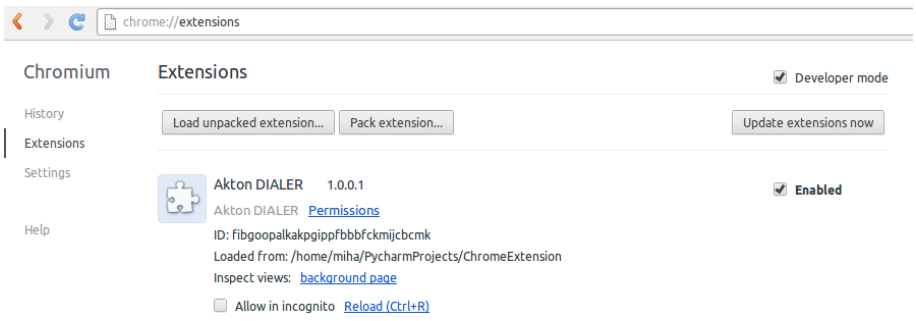
Razširitev Akton DIALER ima unikatno število ID enako "fibgoopalkakpgippfbbbfckmijcbcmk". Ta unikatna zgoščena vrednost ostaja nespremenjena ob uporabi istega ključa pri podpisovanju razširitve, kot smo že omenili v predhodnem poglavju. Po ponovnem grajenju dobimo verzijo spletnega brskalnika Chrome, v kateri deluje dostop do mikrofona tudi iz razširitve Akton DIALER.

Poglavje 4

Rezultat

4.1 Predstavitev razširitve Akton DIALER

Spletni brskalnik Chrome vsebuje posebno servisno stran **chrome://extensions** s seznamom vseh nameščenih razširitev. Nove razširitve se nameščajo s pomočjo mehanizma primi in potegni (drag and drop), razširitveno datoteko CRX preprosto povlečemo na omenjeno stran in razširitev je nameščena. Slika 4.1 prikazuje razširitev Akton DIALER znotraj servisne strani.



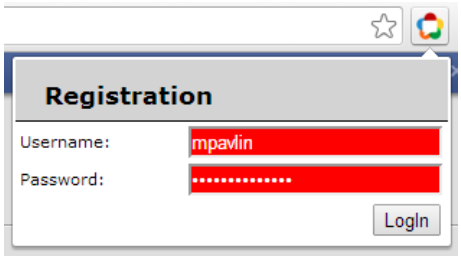
Slika 4.1: Nameščena razširitev Akton DIALER.

Po uspešni namestitvi se na koncu naslovne vrstice spletnega brskalnika prikaže ikona razširitve (slika 4.2). Uporabili smo standardno ikono platforme WebRTC, pet barvnih krogov s sporočilnim oknom v sredini. Ob kliku

nanjo se odpre prijavno okno za vnos domenskega uporabniškega računa. Uporabniški vmesnik je v angleškem jeziku, da je razumljiv vsem zaposlenim znotraj skupine Akton, ki ima podružnice tudi zunaj Slovenije.

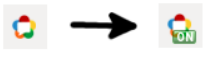
Slika 4.2: Prijavno okno razširitve.

Uporabniško ime se shrani v lokalno bazo brskalnika (chrome.storage), da je naslednja prijava hitrejša. Ob vneseni nepravilni kombinaciji uporabniškega računa in gesla, se vnosni polji obarvata rdeče, kot prikazuje slika 4.3.



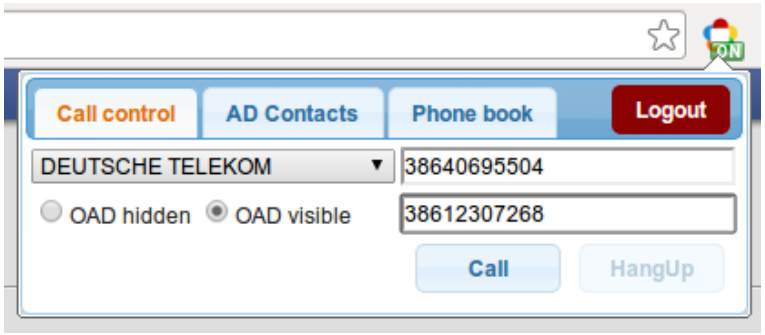
Slika 4.3: Neuspešna prijava.

Po uspešni prijavi se spremeni ikona razširitve, čez njo se izriše besedilo "ON". Tako uporabniku sporočimo uspešnost prijave na strežnik Asterisk in razširitev je pripravljena na klicanje.



Slika 4.4: Sprememba ikone ob uspešni prijavi.

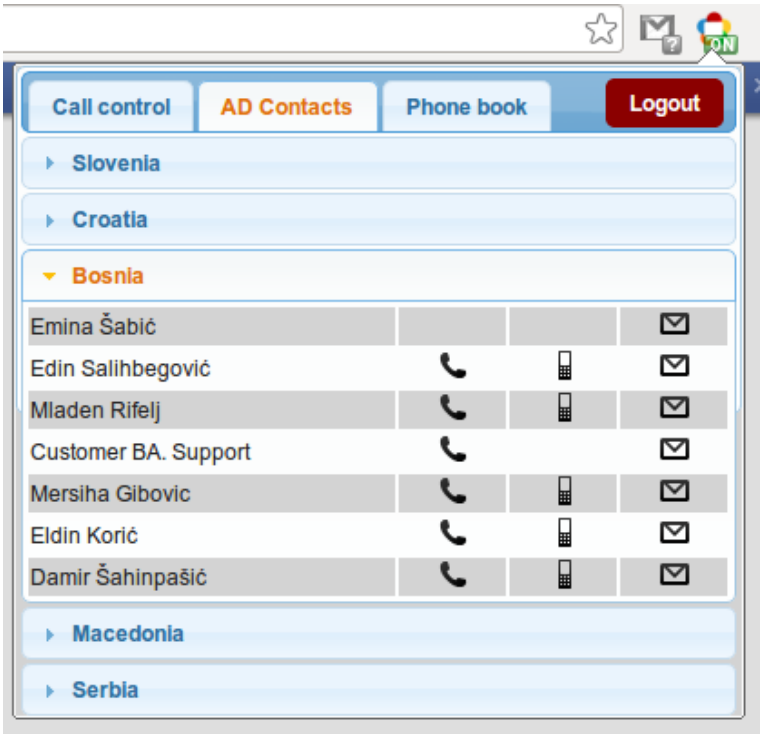
Avtomatično se odpre osrednje okno razširitve (slika 4.5), ki je razdeljeno na več celot, ločeno z zavihki. Na osnovnem oknu je tudi gumb **Logout**, s katerim se odjavimo iz strežnika Asterisk, ikona v naslovni vrstici se povrne v prvotno stanje in prikaže se prijavno okno. Pod zavihki imamo izbirni menu, kjer izberemo način klicanja.



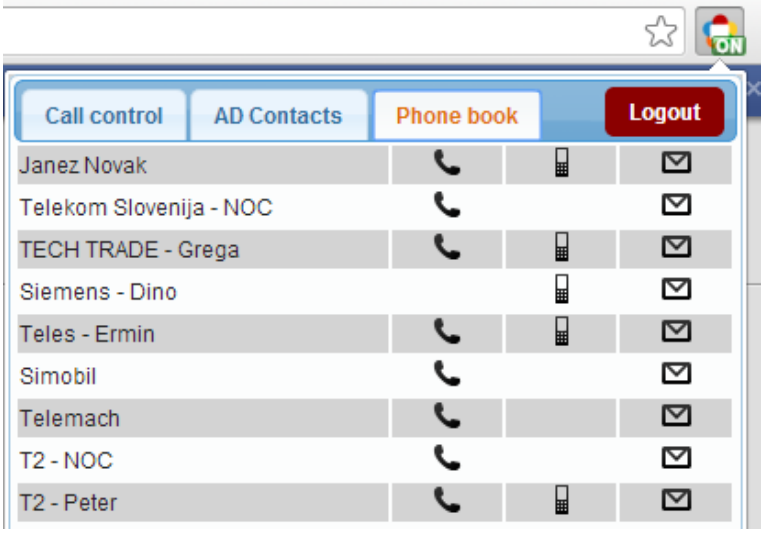
Slika 4.5: Akton Dialer

Prva možnost je izbira usmerjanje preko najcenejšega povezovalnika, kjer nas ne zanima, kateri operater se bo uporabil v procesu. Pod izbiro usmerjanja LCR je seznam vseh operaterjev z usmerjevalnimi kodami. Desno je polje za vnos klicane številke, ki mora biti obvezno v mednarodnem formatu. Sledi izbira skrivanja oziroma prikazovanja identifikacijske številke (OAD - Originating Address), s katero se bomo predstavili klicani osebi. Za identifikacijsko številko prav tako velja pravilo, da mora biti vnešena v mednarodnem formatu. Sledita še gumba **Call** in **Hangup** za vzpostavitve klica in njegovo rušenje.

V zavihku **AD Contacts** (slika 4.6) razširitev prikazuje seznam vseh domenskih uporabnikov skupine Akton. Vsak stik ima polje za stacionarno in mobilno številko ter elektronski naslov. Ob kliku na ikono telefonske številke se izbrana številka avtomatično prenese v polje za izbiro klicane številke znotraj prvega zavihka. Ikona elektronskega poštnega naslova zažene privzeti odjemalec, ki je definiran znotraj operacijskega sistema.



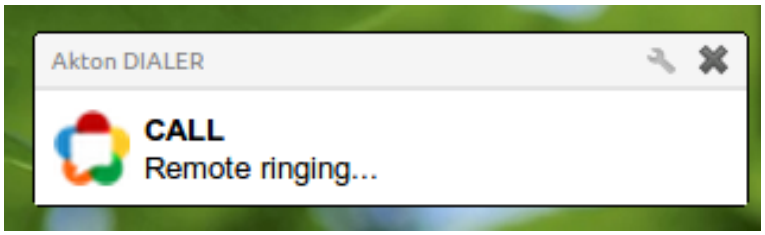
Slika 4.6: Seznam domenskih uporabnikov.



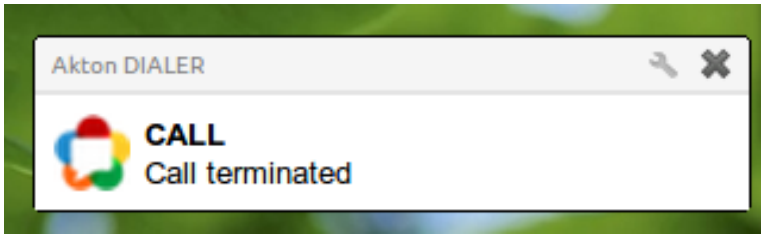
Slika 4.7: Uporabnikovi osebni stiki.

Tretji zavihek "Phone book"(slika 4.7) vsebuje uporabnikove stike, ki jih posameznik ureja znotraj intranet portala.

Razširitev Akton DIALER prikazuje obvestila uporabniku s pomočjo aplikativnega vmesnika **chrome.notifications**. Obvestilo se prikaže v obliki pojavnega okna nad sistemsko opravilno vrstico, sliki 4.8 in 4.9 prikazujeta obliko obvestila znotraj operacijskega sistema Linux za odhodni glasovni klic v stanju zvonjenja in prekinitve zveze.



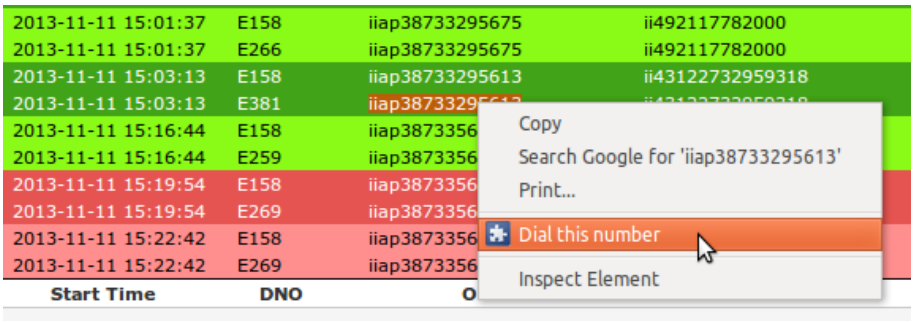
Slika 4.8: Odhodni klic v stanju zvonenja.



Slika 4.9: Obvestilo o prekinitvi zveze.

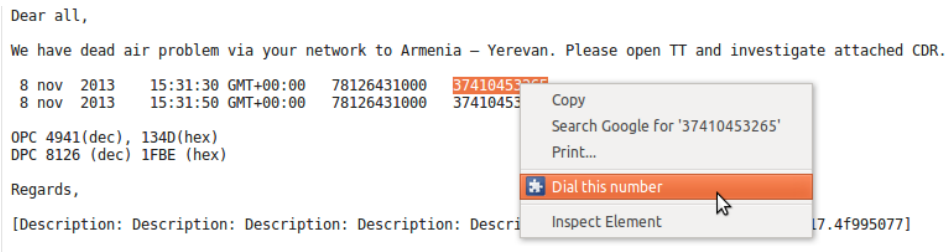
Z desnim klikom znotraj spletne strani v brskalniku Chrome prikažemo kontekstni izbirni meni (context menu). Razširitev Akton DIALER doda možnost hitrega klicanja (slika 4.10), ki označeno telefonsko številko prenese v polje za vnos številke (slika 4.6) in sproži odhodni klic. Možnost hitrega klica je aktivirana po uspešni registraciji na strežnik Asterisk. Znotraj podjetja Akton se s telefonskimi številkami srečujemo na vsakem koraku, zato nam možnost hitrega klicanja olajša vsakodnevne procese pri uporabi internih spletnih aplikacij, kot so:

- Pregledovanje zgodovine klicev znotraj intranet portala (slika 4.10).



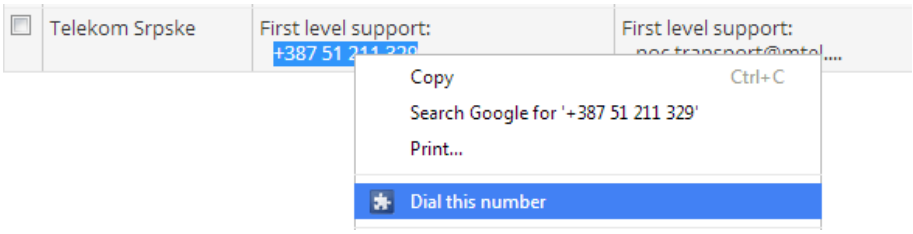
Slika 4.10: Pregledovanje zgodovine klicev.

- Reševanje prijavljenih težav znotraj sistema za prijavo napak. Slika 4.11 prikazuje prejeto sporočilo o napaki našega glasovnega partnerja, v katerem se pritožuje nad problemom slišnosti. Problematično telefonsko številko lahko hitro pokličemo neposredno iz spletnega brskalnika.



Slika 4.11: Branje sporočila znotraj sistema za prijavo napak.

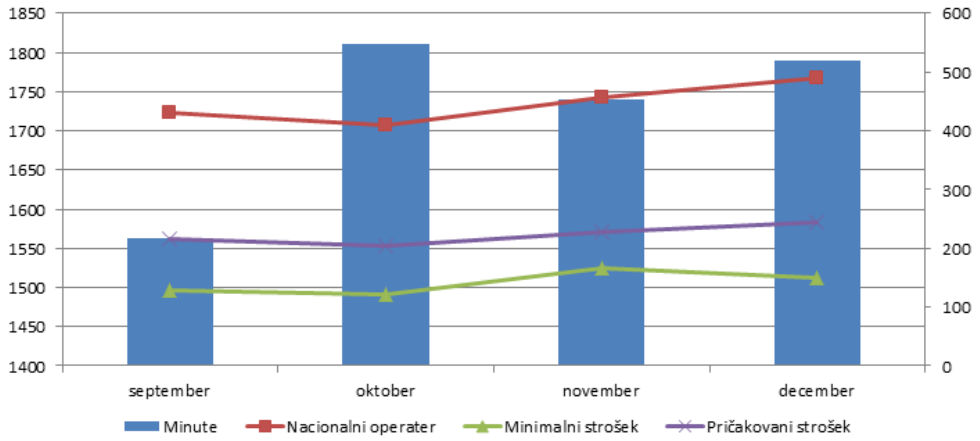
- V primeru težav na podatkovnem/glasovnem vodu, lahko ponudnika storitev hitro pokličemo neposredno iz seznama stikov znotraj dokumentnega sistema Alfresco, kot prikazuje slika 4.12.



Slika 4.12: Seznam kontaktov v dokumentnem sistemu Alfresco.

4.2 Izkušnje

Zaposleni podjetja Akton mesečno opravijo med 1500 in 2000 minut telefonskih pogovorov. Stroški opravljenih klicev se gibljejo med 400 in 500 €, odvisno od klicanih mednarodnih destinacij. Polovica klicev je opravljena na nacionalne številke, četrtina v države Evropske unije, preostala četrtina pa večinoma pripada državam bivše Jugoslavije. Potrebno je omeniti, da nacionalni operater uporablja minutni obračunski interval, vsi glasovni partnerji podjetja Akton pa sekundnega. Že na račun obračunskega intervala so možni večji prihranki.



Slika 4.13: Graf stroškov

Graf 4.13 prikazuje količino minut v zadnjih štirih mesecih leta 2013. Rdeča črta prikazuje trenutne stroške telefonskih pogovorov, opravljenih preko omrežja PSTN nacionalnega operaterja. Zelena črta predstavlja idealni scenarij, če bi bili vsi klici opravljeni preko glasovnih povezav na centrali Teles. Zavedamo se dejstva, da je navada železna srajca, in vsi zaposleni ne bodo uporabljali predlagane razširitve. Vijolična črta predstavlja pričakovane stroške, če bi 75% zaposlenih uporabljalo predlagano rešitev. Pričakovano znižanje stroškov telefonskih pogovorov tako znaša 50%, brez dodatnih investicij podjetja Akton.

Poglavje 5

Sklepne ugotovitve

Z analiziranjem vsakodnevnih procesov v telekomunikacijskem podjetju Akton smo ugotovili pomembnost glasovnega komuniciranja. Pri izdelavi rešitve diplomske naloge smo z uporabo platforme WebRTC razvili rešitev za udobnejšo in cenejšo glasovno komunikacijo. Podrobno smo opisali osrednjo temo diplomskega dela, platformo WebRTC, in ostale tehnologije uporabljene na strani odjemalca in strežnikov. Rezultat diplomskega dela je aplikativna razširitev spletnega brskalnika Chrome, ki omogoča popolno integracijo znotraj obstoječih spletnih aplikacij. Podjetju Akton smo omogočili znižanje stroškov glasovnega komuniciranja in večjo učinkovitost zaposlenih.

Pri razvoju smo naleteli na težavo pri dostopu do mikrofona znotraj razširitve spletnega brskalnika Chrome. Uporaba razširitve Akton DIALER je trenutno mogoča samo znotraj posebne verzije spletnega brskalnika Chrome, ki vsebuje spremenjeno izvorno kodo. V kolikor omejitev dostopa ne bo rešena v prihodnjih verzijah brskalnika Chrome, bo treba razviti alternativno rešitev. Ena od možnosti je preselitev odjemalca WebRTC iz razširitve v običajno spletno stran znotraj portala Akton, razširitev Akton DIALER pa posreduje zahtevke omenjeni spletni strani.

Obstoječo razširitev Akton DIALER bi lahko dopolnili s prikazovanjem statusa prisotnosti (presence) ostalim uporabnikom, ki so v danem trenutku registrirani na strežnik Asterisk. Naslednja stopnja bi bila tekstovno

sporočanje med registriranimi uporabniki znotraj platforme WebRTC, kot je bilo prikazano v drugem poglavju. Dobrodošla bi bila tudi informacija o dolžini trenutno aktivnega telefonskega pogovora.

Literatura

[1] Chromium, "Build instructions for Linux", 2014, https://code.google.com/p/chromium/wiki/LinuxBuildInstructions

[2] Digium, "Get Started", http://www.asterisk.org/

[3] Doubango Telecom, "SIPML5 API documentation", 2012, http://sipml5.org/docgen/index.html?svn=200

[4] Google, 2013, https://codereview.chromium.org/23454030/

[5] Google, "Developer's Guide", 2013, http://developer.chrome.com/

[6] Jonathan Rosenberg,"Interactive Connectivity stablishment: ICE", http://unina.stidue.net/Applicazioni Telematiche/Materiale/ice-basic-tutorial.pdf

[7] Rosenberg, J., Weinberger, J., Huitema, C., Mahy, R., "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", IETF, RFC3489, March 2003, http://ietf.org

[8] The International Engineering Consortium, "Network Computer-Telephony Integration (CTI) Delivering Intelligent Network (IN) Services" http://mktelecom.50webs.com/pdf/net_cti.pdf

[9] Tsahi Levent,"What is WebRTC? ", 2012, http://bloggeek.me/webrtc/

[10] Sam Dutton, "Getting Started with WebRTC", 2012, http://www.html5rocks.com/en/tutorials/webrtc/basics/

[11] TELES AG, "C4 Softswitch Signaling Manual", 2013, C4_SSW_Signaling_3.4M10001070.pdf, stran 60

[12] Wikipedia, "Class 5 telephone switch", http://en.wikipedia.org/wiki/Class_5_telephone_switch

[13] Wikipedia, “Session Initiation Protocol“, http://en.wikipedia.org/wiki/Session_Initiation_Protocol