



УНИВЕРЗИТЕТ У НОВОМ САДУ ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД
Департман за рачунарство и аутоматику
Одсек за рачуарску технику и рачуарске комуникације

ЗАВРШНИ (BACHELOR) РАД

Кандидат: Александар Спасојевић

Број индекса: Е12839

Тема рада: Проширење SIP VoIP клијента са додатном обрадом звука

Ментор рада: Пап др. Иштван

Нови Сад, Септембар, 2012



КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:			
Идентификациони број, ИБР:			
Тип документације, ТД:	Монографска документација		
Тип записа, ТЗ:	Текстуални штампани материјал		
Врста рада, ВР:	Завршни (Bachelor) рад		
Аутор, АУ:	Александар Спасојевић		
Ментор, МН:	Пап др. Иштван		
Наслов рада, НР:	Проширење SIP VoIP клијента са додатном обрадом звука		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2012		
Издавач, ИЗ:	Ауторски репринт		
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО: (поглавља/страна/цитата/табела/слика/графика/прилога)	7/40/0/0/20/0/0		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	SIP протокол, интеграција додатних функција за обраду података у постојећу клијентску апликацију		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У овом раду приказан је један од начина интеграције додатне обраде података у постојећу SIP клијентску апликацију.		
Датум прихватања теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	проф. др. Јелена Ковачевић	
	Члан:	проф. др. Илија Башичевић	Потпис ментора
	Члан, ментор:	проф. др. Иштван Пап	



KEY WORDS DOCUMENTATION

Accession number, ANO:			
Identification number, INO:			
Document type, DT:	Monographic publication		
Type of record, TR:	Textual printed material		
Contents code, CC:	Bachelor Thesis		
Author, AU:	Aleksandar Spasojević		
Mentor, MN:	PhD Pap Išvan		
Title, TI:	Extension of SIP VoIP client with additional voice processing		
Language of text, LT:	Serbian		
Language of abstract, LA:	Serbian		
Country of publication, CP:	Republic of Serbia		
Locality of publication, LP:	Vojvodina		
Publication year, PY:	2012		
Publisher, PB:	Author's reprint		
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6		
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/40/0/0/20/0/0		
Scientific field, SF:	Electrical Engineering		
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems		
Subject/Key words, S/KW:	SIP protocol, integration of additional functions for data processing in existing client application		
UC			
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia		
Note, N:			
Abstract, AB:	This work presents one way for integrating additional functions for data processing in existing SIP client application		
Accepted by the Scientific Board on, ASB:			
Defended on, DE:			
Defended Board, DB:	President:	Jelena Kovačević, PhD	
	Member:	Ilija Bašičević, PhD	Mentor's sign
	Member, Mentor:	IšvanPap, PhD	

Zahvalnost

Zahvaljujem se mentoru, prof. dr Ištvanu Papu, na vođenju, stručnim i ličnim savetima prilikom izrade ovog rada.

Takođe želim da se zahvalim porodici, kolegama i prijateljima na podršci ne samo tokom izrade ovog rada, već i tokom celokupnog vremena provedenog na studijama.

SADRŽAJ

1.	Uvod	1
2.	Teorijske osnove.....	3
2.1	SIP (<i>Session Initiation Protocol</i>)	3
2.1.1	Internet pozivi kroz SIP zastupnika.....	3
2.1.2	Pozivi ka standardnim telefonskim brojevima	4
2.2	<i>Linphone</i> klijentska aplikacija	4
2.2.1	Unutrašnja struktura <i>Linphone</i> klijentske aplikacije	4
2.2.1.1	<i>mediastreamer2</i> – biblioteka za rad sa multimedijalnim tokovima podataka	5
2.2.1.2	Princip rada <i>mediastreamer2</i> biblioteke	6
2.2.1.3	oRTP (RTP RFC3550) biblioteka.....	6
2.2.1.4	Proširena „osip“ biblioteka	7
3.	Koncept rešenja	8
3.1	<i>Medistreamer2</i>	8
3.1.1	Tok podataka	11
3.2	“seam” filter	12
3.3	Korisnička sprega (UI).....	13
4.	Programsko rešenje.....	15
4.1	<i>Medistreamer2</i> – proširenja	15
4.1.1	Struktura koja predstavlja stanje filtera (struct SeamState) – proširenja	15
4.1.2	Funkcija za inicijalizaciju polja iz structure stanja (seam_init)	17
4.1.3	Funkcija za oslobođanje polja iz strukture stanja (seam_uninit).....	17
4.1.4	Funkcija za predobradu (seam_preprocess)	17
4.1.5	Funkcija za procesiranje (seam_process)	18
4.1.6	Funkcija za krajnju obradu (seam_postprocess).....	18
4.1.7	Funkcija za postavljanje zastavice za audio lokalizaciju (seam_set_audio_localization)	18
4.2	Korisnička sprega (UI) – proširenja.....	19
4.2.1	Struktura koja predstavlja mogućnost “libseamprocessing.so” biblioteke.....	19
4.2.2	Funkcija za pribavljanje statusa o izabranim opcijama (GetMedia2Caps)	20

4.2.3	Funkcija za pribavljanje statusa o izabranim opcijama (GetSeamCaps).....	20
4.2.4	Funkcija za inicijalizaciju parametara za biranje obrade (initCaps)	20
4.2.5	Funkcija za iscrtavanje izabranih opcija (linphone_gtk_init_m2_engine_caps)	21
4.2.1	Funkcija za iscrtavanje izabranih opcija (linphone_gtk_init_o_engine_caps)..	21
4.2.2	Funkcija za obradu zahteva za poništavanje eha (linphone_gtk_echo_cancelation_toggled).....	22
4.2.3	Funkcija za postavljanje zastavice za izabranu obradu (EngineEchoStatus)	22
4.3	“coreapi”- proširenja.....	23
4.3.1	Funkcija za pokretanje prenosa podataka (audio_stream_start_full)	23
4.3.2	Funkcija za postavljanje zastavice za biranje obrade (linphone_core_enable_seam_echo_cancellation)	24
4.3.3	Funkcija za očitavanje postavljene obrade (linphone_core_echo_cancellation_enabled)	25
5.	Rezultati.....	26
5.1	Potiskivanje šuma	26
5.2	Automatska kontrola jačine zvuka.....	27
5.3	100Hz filtriranje.....	28
5.4	Prostorno filtriranje i audio lokalizacija	29
5.5	Poništavanje eha.....	29
6.	Zaključak	31
7.	Literatura	32

SPISAK SLIKA

Slika 2.1 – Programska arhitektura aplikacije	5
Slika 2.2 – Lanac filtera.....	6
Slika 3.1 – Dijagram zavisnosti <i>_MSFilter</i> objekta.....	9
Slika 3.2 – Graf za slanje podataka.....	10
Slika 3.3 – Graf za prijem podataka	10
Slika 3.4 – Graf za slanje podataka uz „seam“ proširenje	11
Slika 3.5 – Graf za prijem podataka uz “seam” proširenje.....	12
Slika 3.6 – Korisnička sprega	14
Slika 5.1 – Signala sa šumom	26
Slika 5.2 – Signal sa potisnutim šumom.....	27
Slika 5.3 – Signal bez primenjene automatske kontrole jačine zvuka.....	27
Slika 5.4 Signal sa primenjenom automatskom kontrolom jačine zvuka	27
Slika 5.5 – Signal bez primjenog filtriranja na 100Hz.....	28
Slika 5.6 – Signal sa primjenjenim filtriranjem na 100Hz.....	28
Slika 5.7 – Testni signal.....	29
Slika 5.8 – Izlazni signal posle primjenjenog algoritma za prostorno filtriranje	29
Slika 5.9 – Ulazni signal koji sadrži echo	30
Slika 5.10 – Echo signal koji treba ukloniti.....	30
Slika 5.11 – Izlazni signal sa uklonjenim ehom	30

SKRAĆENICE

- SIP** - *Session Initiation Protocol*, Protokol za kontrolisanje komunikacionih sesija
- UI** - *User Interface*, Korisnička sprega
- RTP** - *Real-time Transfer Protocol* – Protokol za prenos podataka u realnom vremenu
- IETF** - *Internet Engineering Task Force*, Organizacija za razvijanje i promovisanje internet standarda
- HTTP** - *Hyper Text Transfer Protocol*, Osnovni internet protokol za slanje datoteka
- TCP** - *Transmission Control Protocol*, Protokol za praćenje prenosa podataka
- UDP** - *User Datagram Protocol*, Protokol za prenos datagrama
- RFC** - *Request For Comments*, Predstavlja zvaničnu specifikaciju internet standarda
- SMTP** - *Simple Mail Transfer Protocol*, Protokol za elektronsku poštu
- SCTP** - *Stream Control Transmission Protocol*, Protokol za praćenje tokova podataka
- IP** - *Internet Protocol*, Internet protokol
- SDK** - *Software Development Kit*, Alat za razvoj aplikacija
- API** - *Application Programming Interface*, Sprega za komunikaciju programskih komponenti
- VoIP** - *Voice over IP*, Protokol za prenos audio i video tokova podataka
- PSTN** - *Public Switched Telephone Network*, Mreža javnih telefonskih mreža
- PCM** - *Pulse-code Modulation*, Metoda za digitalno predstavljanje analognog signala

1. Uvod

U ovom radu opisan je način integracije biblioteke za obradu govornog signala u postojeću SIP aplikaciju. Opisan je i način funkcionisanja SIP protokola koji kontroliše komunikacione zadatke kao što je prenos videa i zvuka između dva učesnika. Polazna tačka rada je program za komunikaciju *Linphone* koji je otvorenog koda. Rešenje je realizovano u programsку jeziku C, na Linux operativnom sistemu na kome je i testirano, mada se uz manje promene može upotrebiti i na Windows operativnom sistemu kao i na Android platformi.

Realizovano rešenje treba da obezbedi dodatnu audio obradu pored podrazumevane u *Linphone*, kao što je npr. potiskivanje eha, kao i potrebnu korisničku spregu (eng. User Interface - UI) za kontrolu obrade. Dodatni algoritmi za obradu govornog signala se nalaze u dатој biblioteci *seam*. Pre uspostavljanja veze bira se jedan od raspoloživih algoritama za obradu (npr. eliminisanje eha) iz postojećih biblioteka, i on se primenjuje u realnom vremenu za obradu zvuka. Pošto prenos audio i video podataka zahteva postojanje RTP protokola i on će biti dodatno objašnjen, kao i njegova veza sa SIP protokolom.

Rad sadrži sedam poglavlja.

U prvom poglavlju je opisan sadržaj rada i sam zadatak.

Drugo poglavlje opisuje teoretske osnove prenosa podataka posredstvom SIP protokola, kao i osnove RTP protokola. U okviru opisa *Linphone* klijentske aplikacije opisana je i *mediastreamer2* biblioteka koja sadrži funkcije za obradu video i audio tokova podataka. Ukratko je opisana i GTK+ sprega za realizaciju rešenja korisničke sprege.

U trećem poglavlju opisan je koncept rešenja. Pre svega su detaljno opisani delovi aplikacije koji učestvuju u realizaciji rešenja a potom i sam koncept rešenja.

Četvrto poglavlje sadrži opis programskog rešenja. Navedene su sve funkcije koje čine rešenje, njihov prototip I parametri uz kratko objašnjenje.

Peto poglavlje opisuje rezultate realizovane obrade. Rezultati u ovom slučaju predstavljaju kvalitet rada algoritama za specifičnu obradu, konkretno kvalitet zvuka koji se dobija na izlazu. Verifikacija se može vršiti ili slušanjem rezultata obrade pri čemu se na ulaz algoritama dovodi poznat signal, ili posmatranjem spektralnih ili frekvencijskih karakteristika signala.

Šesto poglavlje sadrži zaključke o realizovanom zadatku i dobijenim rezultatima kao i mogućim poboljšanjima rešenja.

Sedmo poglavlje predstavlja spisak korišćene literature.

2. Teorijske osnove

2.1 SIP (*Session Initiation Protocol*)

SIP predstavlja protokol za uspostavljanje poziva, odnosno prenos govora preko IP (Internet Protocol) protokola. To je otvoreni standard, objavljen od strane IETF organizacije. Inspirisan je na osnovu protokola za elektronsku poštu i HTTP-a. Protokol se može koristiti za kreiranje, menjanje i završavanje sesija između dva učesnika (*unicast*) ili više učesnika (*multicast*). Sesije se mogu sastojati od jednog ili više tokova podataka. SIP protokol je na aplikativnom sloju kreiran tako da bude nezavistan od nižeg transportnog sloja. Može se realizovati pomoću TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*) ili SCTP (*Stream Control Transmission Protocol*) protokola. To je protokol baziran na tekstu, uključujući mnogo elemenata iz HTTP (*Hypertext Transfer Protocol*) i SMTP (*Simple Mail Transfer Protocol*) protokola. U nastavku će se opisati osnovne karakteristike SIP-a sa stanovišta korisnika, kao i ostvarivanje poziva preko SIP protokola.

2.1.1 Internet pozivi kroz SIP zastupnika

Na internetu, zbog promenjivih IP adresa, mobilnosti, zaštitnih zidova (*firewalls*), direktni pozivi nisu korisni. To je razlog zbog kog koncept SIP-a postoji: zastupnik je odgovoran za usmeravanje poziva bez obzira koja je IP adresa krajnjeg korisnika. VoIP operateri postavljaju SIP zastupnike na Internet radi usmeravanja poziva svojih korisnika .

Prvi korak je da se korisnik pretplati na željenu uslugu birajući korisničko ime i šifru. Javni identitet korisnika onda postaje <korisničko ime>@<VoIP provajder> i to je adresa koju ostali korisnici koriste da dobiju određenog korisnika preko SIP-a.

Nakon toga, konfiguriranjem klijenta (kao što je *Linphone*) da komunicira kroz zastupnika, dobija se javna SIP adresa. Na isti način se dobija adresa za elektronsku poštu. Drugi korisnici koji su preplaćeni na istog operatera kontaktiraju preko korisničkog imena ili broja telefona. Prosleđivanje Internet poziva je usluga koja je najčešće besplatna.

2.1.2 Pozivi ka standardnim telefonskim brojevima

VoIP operateri često omogućavaju VoIP ka PSTN (klasična telefonija) pozive po povoljnim cenama kroz njihove zastupnike. Na taj način mogući su pozivi prema bilo kome u svetu pomoću VoIP jednostavnim unošenjem telefonskog broja. SIP korisničko ime postaje telefonski broj.

2.2 *Linphone* klijentska aplikacija

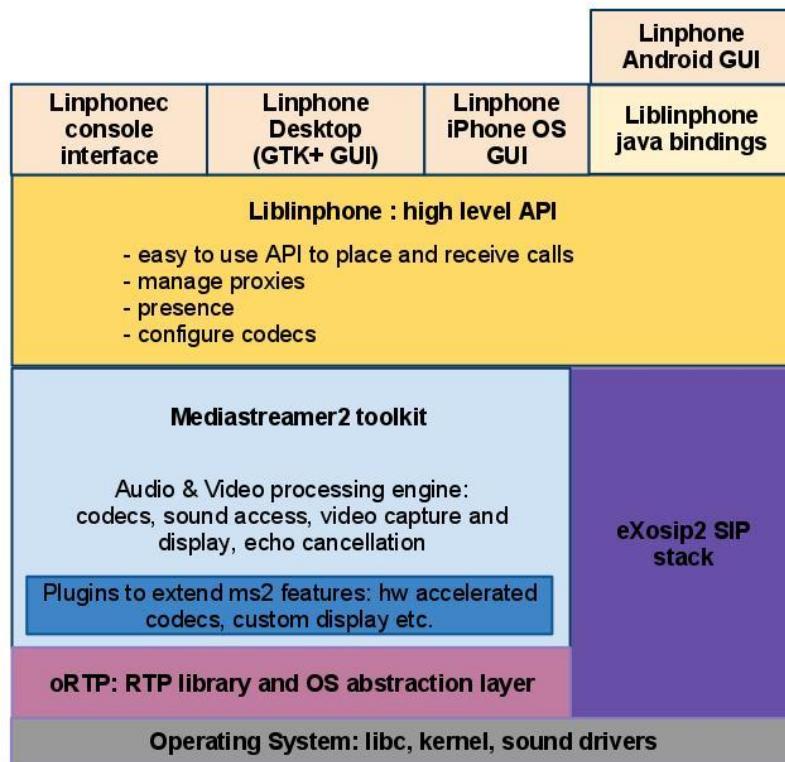
Linphone predstavlja aplikaciju za telefoniranje preko interneta. Pomoću ove aplikacije moguća je besplatna komunikacija sa bilo kim preko interneta, i to video, govorna, ili dopisivanje. Aplikacija koristi SIP protokol, otvoreni standard za Internet telefoniju.

2.2.1 Unutrašnja struktura *Linphone* klijentske aplikacije

Unutrašnja struktura aplikacije je razdvojena između korisničke sprege i glavnog sistemskog dela, dozvoljavajući kreiranje raznih korisničkih sprega za istu funkcionalnost.

- Korisničku spregu predstavljaju:
 - Gtk+/Glade sprega
 - Konzolna sprega
 - iPhone aplikacija implementirana u objektno orijentisanom C-u
 - Android aplikacija realizovana u Java jeziku
- Osnovni deo aplikacije: ovo je biblioteka koja realizuje osnovnu funkcionalnost aplikacije. Pruža API visokog nivoa za iniciranje, primanje i završetak poziva. Oslanja se na sledeće programske komponente:
 - Mediastreamer2, moćan multimedijalni SDK (*Software Development Kit*) za pravljenje audio/video tokova podataka i njihovu obradu.
 - ORTP, RTP (*Real Time Protocol*) biblioteka.
 - eXosip2, SIP korisnička biblioteka bazirana na *libosip2*.

Liblinphone i svi njegovi delovi su realizovani u programskom jeziku C.



Slika 2.1 – Programska arhitektura aplikacije

2.2.1.1 **mediastreamer2 – biblioteka za rad sa multimedijalnim tokovima podataka**

Specijalizovana je za prenos slike/zvuka u aplikacijama za telefoniranje. Ova biblioteka je odgovorna za slanje i primanje multimedijalnih podataka u linphone-u, uključujući snimanje zvuka/slike, kodovanje i dekodovanje, kao i prikaz.

Ono što *mediastreamer2* omogućava jeste:

- Čitanje/pisanje sa/na “alsa” audio uređaje, “oss” uređaje, “waveapi” uređaje
- Slanje i prijem RTP paketa
- Kodovanje i dekodovanje određenih formata zvuka i slike
- Čitanje i pisanje sa/u “wav” datoteke
- Čitanje i prikazivanje YUV slika sa kamere
- Poništavanje eha

- Konferencijski pozivi
- Ekvilajzer korišćenjem FIR filtera
- Kontrola jačine zvuka

2.2.1.2 Princip rada *mediastreamer2* biblioteke

Svaki deo obrade je sadržan u *MSFilter* objektu. *MSFilter* objekti imaju ulaze I/ili izlaze koji se koriste za povezivanje vise *MSFilter* objekata. U nastavku je dat jednostavan primer koji opisuje kako povezivanje funkcioniše:

- *MSRtpRecv* je *MSFilter* koji prima RTP pakete sa mreže, raspakuje ih i prosleđuje na svoj izlaz.
- *MSSpeexDec* je *MSFilter* koji uzima sve sa svog ulaza i prepostavljaći da su to paketi kodovani na njemu odgovarajući način, dekoduje ih i prosleđuje na svoj izlaz.
- *MSFileRec* je *MSFilter* koji uzima sve sa svog ulaza i to zapisuje u “wav” datoteku.

MSFilteri mogu biti povezani tako da oforme lanac filtera. Ako posmatramo gore opisan postupak i povežemo filtere dobićemo lanac koji prima RTP pakete, dekoduje ih i potom zapisuje u “wav” datoteku.



Slika 2.2 – Lanac filtera

Lanac obrade multimedijalnih podataka pokreće *MSTicker* objekat, nit koja se budi svakih 10ms da obradi podatke koji se nalaze u lancu filtera kojima rukovodi. Nekoliko *MSTicker* objekata se mogu koristiti uporedno, na primer jedan za audio filtere, jedan za video filtere, ili jedan za svaki procesor na mašini na kojoj je aplikacija pokrenuta.

2.2.1.3 oRTP (RTP RFC3550) biblioteka

Funkcionalnost u oRTP biblioteci:

- Pisana u C-u, radi pod Linux i Windows operativnim sistemima.
- Uključuje podršku za više profila, AV profil je podrazumevani.

-
- Uključuje raspoređivač paketa, za slanje i primanje paketa „na vreme“, u skladu sa njihovom vremenskom oznakom. Raspoređivanje je opcionalno.
 - Uključuje multipleksiranje IO, tako da stotine RTP sesija mogu biti zakazane u isto vreme.
 - Podržava deo RFC2833 za telefonske događaje preko RTP.

2.2.1.4 Proširena „osip“ biblioteka

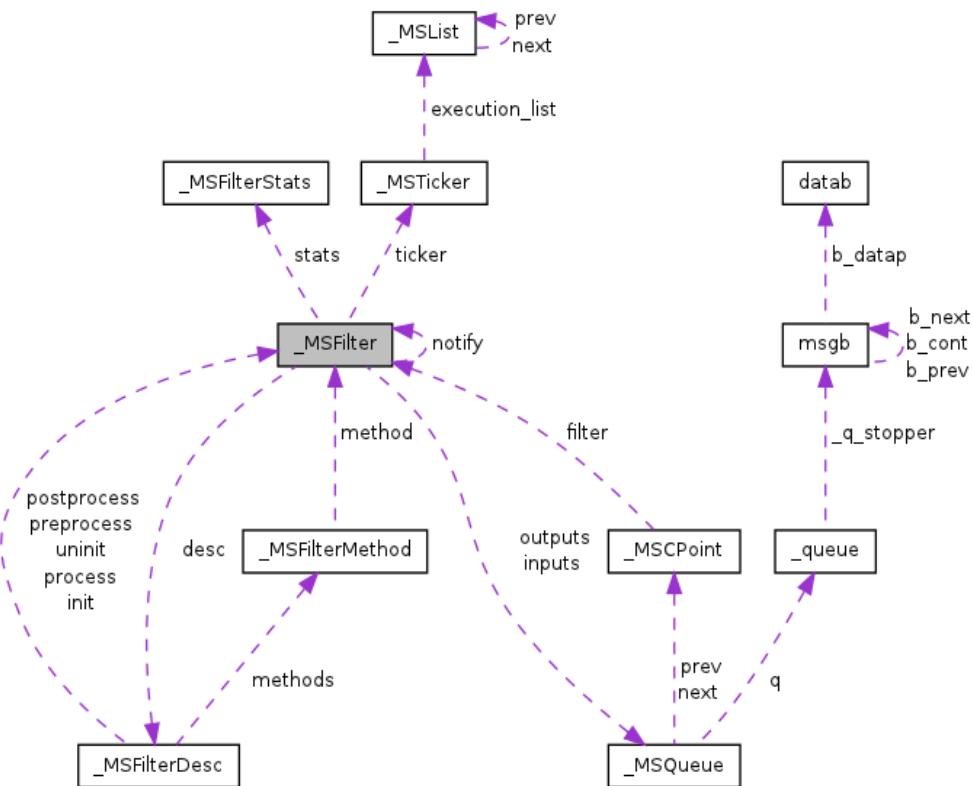
eXosip je biblioteka koja skriva kompleksnost korišćenja SIP protokola za uspostavljanje multimedijalnih sesija. Ovaj protokol se prvenstveno koristi za VoIP telefoniju ali takođe može da bude koristan za aplikacije koje žele da uspostave veze kao u višekorisničkim igrama.

3. Koncept rešenja

Sama obrada audio i video podataka je sadržana u mediastreamer2 biblioteci, pa je stoga osnovna proširenja potrebno pre svega uneti u ovu biblioteku a kasnije se podrška korisničkim interfejsom dodaje kroz GTK+/Glade spregu.

3.1 Mediastreamer2

Mogućnosti ove biblioteke su navedene u teoretskim osnovama pa ćemo ovde samo ukratko opisati način funkcionalnosti onih delova koji su od važnosti za planirano proširenje . Sama obrada audio i video podataka se obavlja kroz objekte koji se nazivaju *MSFilter*. Ideja je da se napravi objekat tipa *MSFilter* i da se u njega smesti željena obrada. Realizovan objekat će se nalaziti neposredno na ulazu grafa procesiranja, nakon filtera koji vrši enkodovanje podataka, odnosno u izlaznom grafu nakon objekta koji vrši dekodovanje podataka.



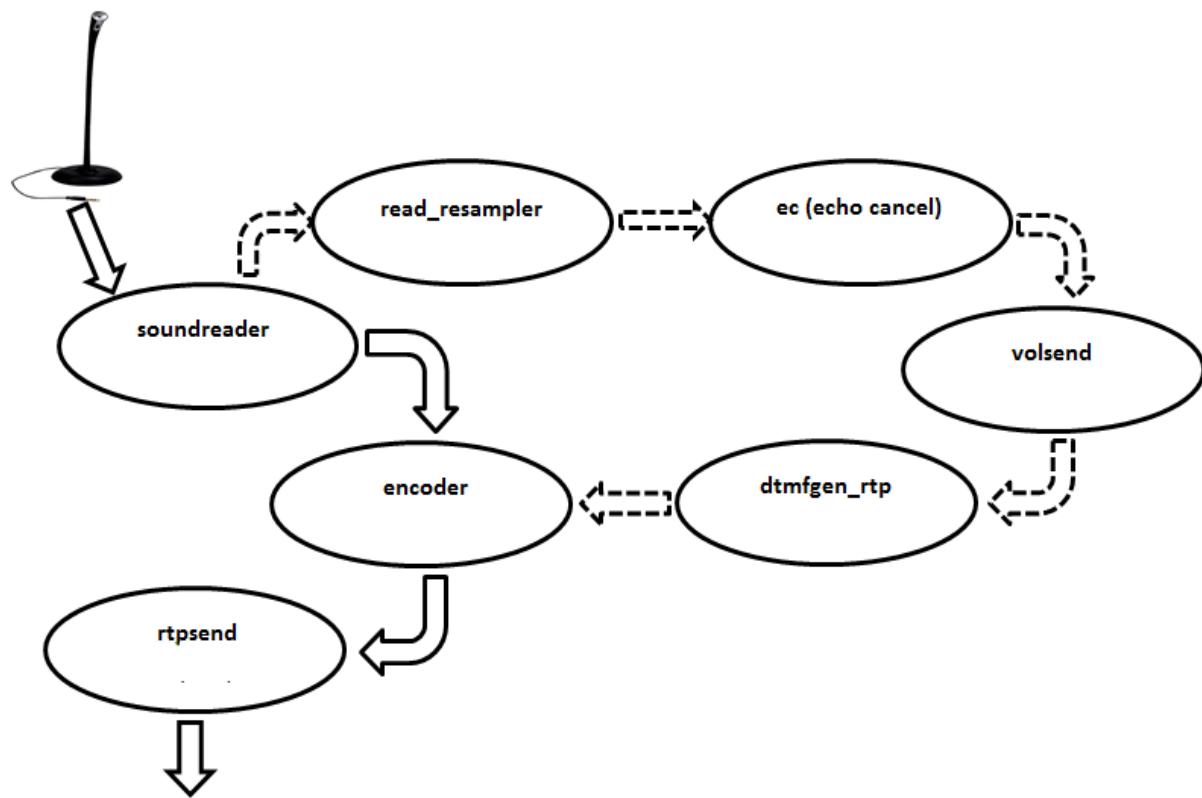
Slika 3.1 – Dijagram zavisnosti `_MSFilter` objekta

Dijagram predstavlja unutrašnju strukturu `_MSFilter` objekta. Na grafu su prikazane osnovne funkcije koje obavljaju obradu. Svaki `_MSFilter` ima istu unutrašnju strukturu. Funkcije koje su u njemu deklarisane mogu a ne moraju da budu implementirane. Moguće je dodati funkcije koje se izvršavaju pri kreiranju filtera ili pri pokretaju grafa.

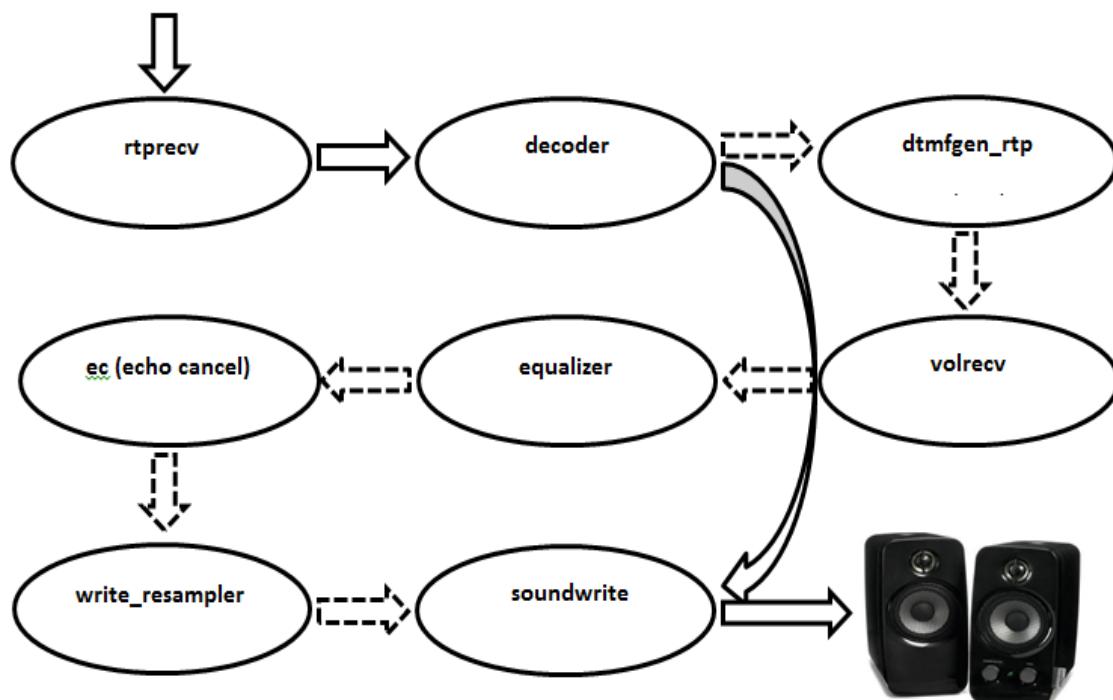
Jednostavan primer osnovnog toka podataka je dat u teoretskim osnovama ali će ovde kroz graf biti naveden tok podataka koji se zapravo nalazi u originalnoj implementaciji. Razlikuju se dva grafa i to:

- Graf za slanje podataka.
- Graf za prijem podataka.

`MSFilter` objekti zajedno povezani formiraju graf i u jednom i u drugom slučaju. Postojeći filter za poništavanje eha se uvek kreira ali se u zavisnosti od parametara uvezuje u graf ili ne. Na sličan način će biti realizovana i dodatna obrada. Nakon formiranja grafa prema datim specifikacijama uz pomoć “ticker” API-ja pokreće se ili zaustavlja graf. U nastavku su prikazani postojeći grafovi koji realizuju osnovnu funkcionalnost.



Slika 3.2 – Graf za slanje podataka

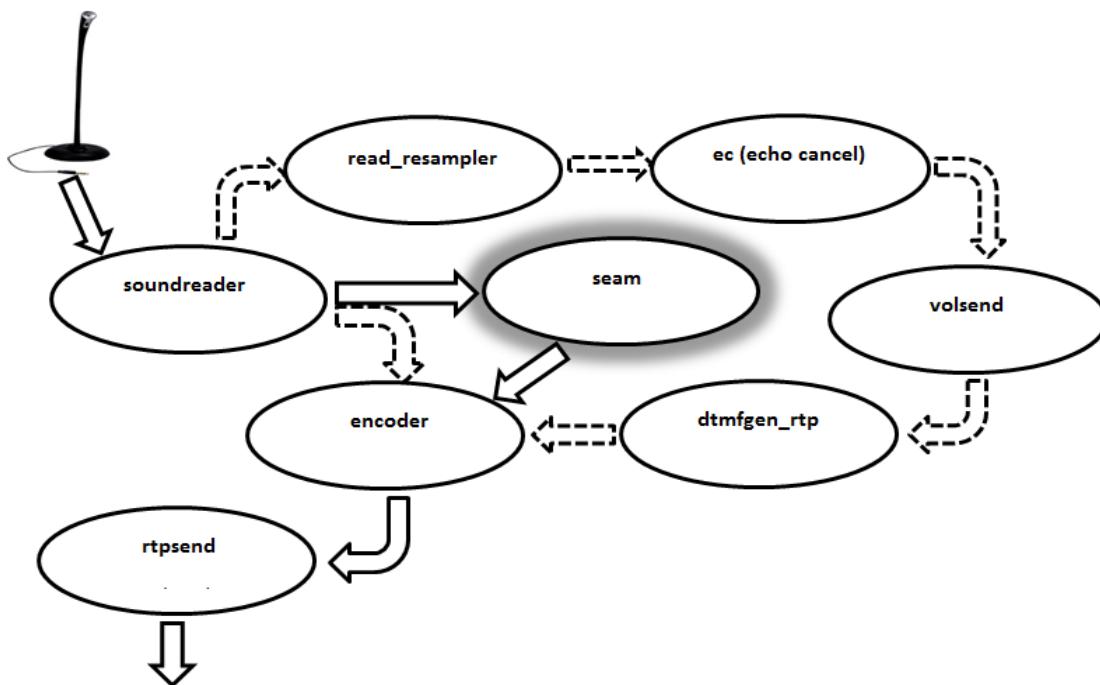


Slika 3.3 – Graf za prijem podataka

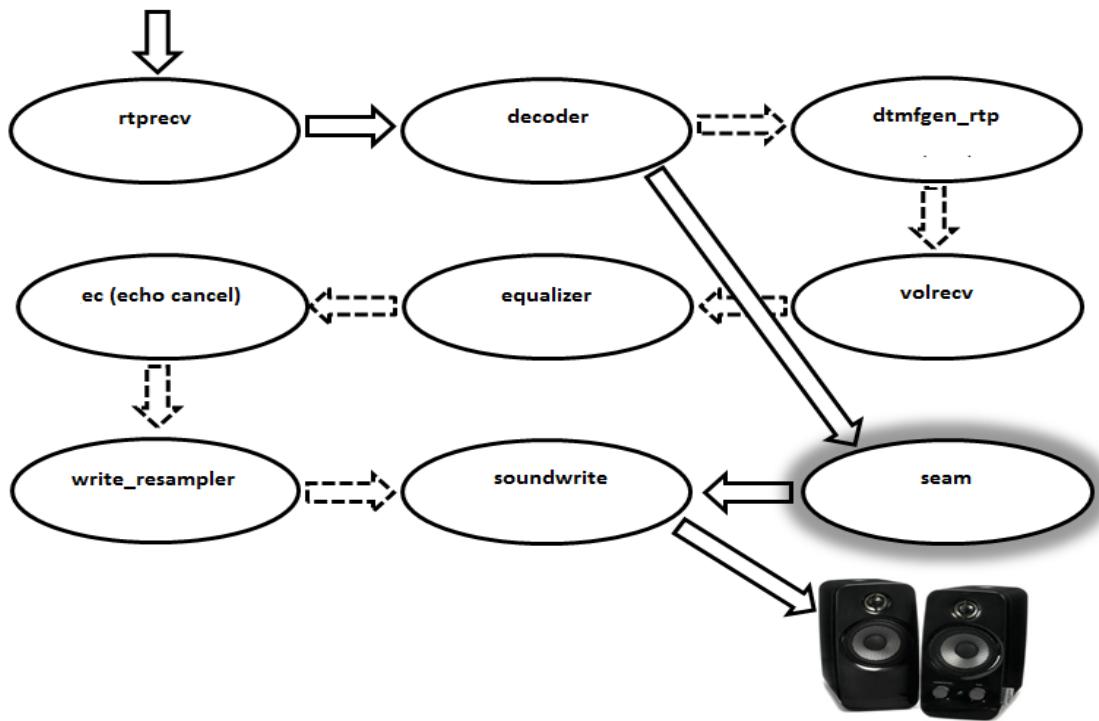
Neće biti opisani detalji realizacije svih komponenti zbog njihove slabe povezanosti sa programskim rešenjem. Potrebno je samo istaknuti da sa prethodnih grafova isprekidane strelice predstavljaju opcione obrade, dok pune predstavljaju tokove kojima će podaci sigurno ići. Ako pratimo tok podataka isprekidanim strelicama putanja nije uslovna. Podaci npr. kod grafa za slanje mogu da idu od „soundreader“ filtera direktno u „ec(echo cancellation)“ filter. Dodatna obrada će biti realizovana neposredno posle čitanja podataka sa mikrofona koje vrši *MSFilter* „soundreader“ objekat i na njegovom izlazu će biti šesnaestobitni neobrađeni odbirci.

3.1.1 Tok podataka

Sa slike 3.2 i 3.3 vidimo da je graf za prijem samo inverzan grafu za slanje. Iz tog razloga ukratko će biti opisan samo graf za slanje podataka uz detaljan opis “seam” filtera. Na sledećim slikama su prikazani isti grafovi ali sa uključenom dodatnom obradom.



Slika 3.4 – Graf za slanje podataka uz „seam“ proširenje



Slika 3.5 – Graf za prijem podataka uz “seam” proširenje

Pri opisu grafa za slanje pratićemo osnovnu putanju jer samo nam je ona bitna za uspešno integrisanje dodatne obrade.

- “soundreader” filter prima podatke sa mikrofona i prosleđuje ih na izlaz u obliku šesnaestobitnih PCM odbiraka.
- “seam” filter u svojoj strukturi sadrži više mogućih obrada pri čemu se željena bira iz korisničke sprege. On prima odbirke iz prethodnog filtera, obrađuje i prosleđuje na svoj izlaz.
- “encoder” filter prima podatke iz prethodnog bloka, koduje i prosleđuje sledećem bloku.
- “rtpsend” filter prima podatke od prethodnog bloka i zadužen je za implementaciju RTP protokola. U isporuci odbiraka ne sme da bude kašnjenja.

3.2 “seam” filter

Kompletna dodatna obrada je sadržana u „seam“ filter objektu. Funkcije za dodatno procesiranje se nalaze u „libseamprocessing.so“ biblioteci pa se ona povezuje u aplikaciju i kroz postojeći API se pristupa njenim funkcijama. Sama procedura obrade iz ove biblioteke mora da ide sledećim tokom:

- Poziva se funkcija za kreiranje konteksta kojoj se prosleđuju učestalost odabiranja, broj kanala i veličina bloka obrade.
- Dobijeni kontekst se zatim prosleđuje funkciji koja bira tip obrade koji će se vršiti nad podacima.
- Nakon toga se poziva sama obrada i dobijeni podaci se prosleđuju sledećem stepenu obrade, u našem slučaju *MSFilter „encoder“* objektu.

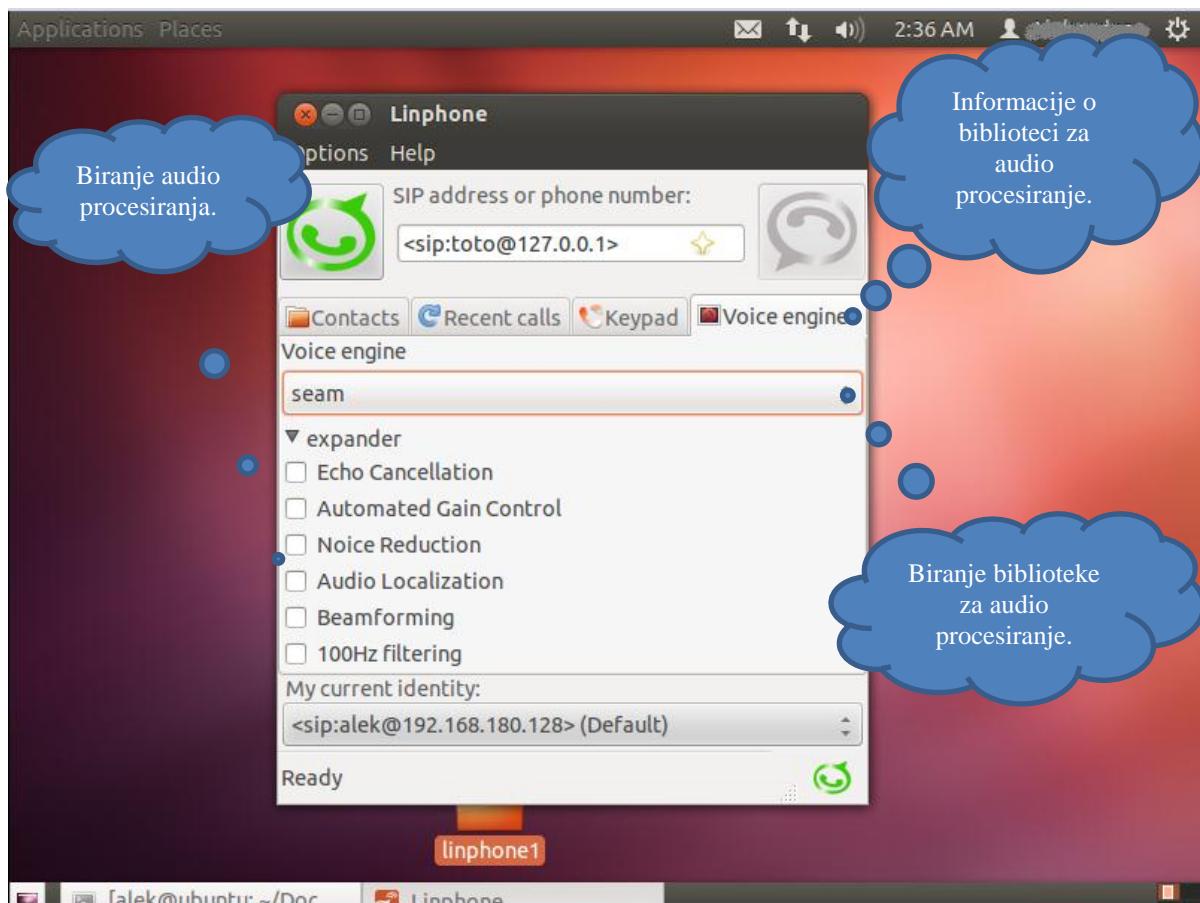
Uz osvrt na sliku 3.1 koja predstavlja dijagram povezanosti *_MSFilter* objekta, određuje se unutrašnja struktura „seam“ filtera. On treba da implementira sledeće funkcije:

- **Init** – inicijalizuje sve promenjive koje su potrebne za predobradu, obradu i finalnu obradu.
- **Uninit** – oslobađa resurse.
- **Preprocess** – inicijalizacija struktura potrebnih za dalju obradu. Pribavljanje vrednosti pojedinih parametara iz prethodnih filtera u grafu.
- **Process** – sama obrada audio podataka.
- **Postprocess** – oslobađanje zauzetih resursa.

„**Process**“ funkcija se izdavja od ostalih jer se u njoj odvija kompletna obrada. Funkcije za obradu audio podataka se samo pozivaju unutar *process* funkcije a nalaze se u „libseamprocessing.so“ biblioteci. Detalji obrade koja se odvija u njima nisu neophodni za krajnje rešenje ali će biti opisan API. Ono što je potrebno jeste da se svi parametri neophodni za pozivanje funkcija za obradu, kao što je učestalost odabiranja, veličina blokova obrade, prilagode pozivanoj funkciji.

3.3 Korisnička sprega (UI)

Prethodno opisana funkcionalnost treba biti podržana korisničkom spregom. Postojeća korisnička sprega će biti proširena korišćenjem Gtk+/Glade sprege.



Slika 3.6 – Korisnička sprega

Proširenja se sastoje od:

- Nov prozor sa informacijama vezanim za audio biblioteke
- Paleta sa izborom audio biblioteke
- Spisak algoritama koje određena biblioteka podržava sa mogućnošću da se neki od njih izabere prostim štikliranjem
- U podmeniju opcija u okviru palete osobina je dodata opcija koja omogućava izbor režima rada sa fajlovima.

Svim vizuelnim proširenjima je kroz Gtk+ spregu dodata funkcionalnost.

4. Programsко rešenje

Proširenje aplikacije je realizovano u programskom jeziku C korišćenjem postojećih API-ja, uz podršku *Linux* gedit teksta editora. Grafičko okruženje je prošireno korišćenjem *Gtk+/Glade* sprege da podržava programsko proširenje. Ne nalazi se u detalje funkcionisanja svih komponenti ali se intenzivno koristi deo *mediastreamer2* API-ja koji se takođe i proširuje, a proširuju se i “core” API kao i API za korisničku spregu.

4.1 *Medistreamer2* – proširenja

U okviru ove biblioteke je realizovana komponenta tipa *MSFilter* pod nazivom “seam”. Unutrašnja struktura je ista kao na slici 3.1. Ova komponenta poseduje sve funkcionalnosti kao i *MSSpeexEC* filter, s tim što su dodata proširenja koja obezbeđuju korišćenje algoritama iz “libseamprocessing.so” biblioteke. U nastavku je dat opis funkcija koje obezbeđuju željenu obradu.

4.1.1 Struktura koja predstavlja stanje filtera (struct **SeamState**) – proširenja

```
typedef struct SeamState
{
    ...
    void* context;
    void* lib_handle;
    void* (*init)(unsigned short _usSampleRate, unsigned short _usInputChannels,
                  unsigned _uiBlockSizePerChannelInSamples);
    BOOL (*set)(void* _pContext, const char* _szAlgorithmParameterName, const
                void* _pAttributeValue);
```

```

BOOL (*get)(void* _pContext, const char* _szAlgorithmParameterName,
            void* _pAttributeValue);
BOOL (*process)(void* _pContext, const short* _psInputSamples,
                short* _psOutputSamples);
BOOL (*analyze)(void* _pContext, const short* _psInputSamples,
                unsigned char* _pucFlagBuffer);
BOOL (*cleanup)(void* _pContext);
BOOL (*reset)(void* _pContext);
bool_t audio_localization;
bool_t echo_cancel;
bool_t noice_reduction;
bool_t agc;
bool_t cbf;
bool_t fil;
}SeamState;

```

Polja:

- **context:** pokazovač na kontekst koji se prosleđuje funkcijama za biranje algoritma i obradu.
- **lib_handle:** ručica otvorene biblioteke.
- **init:** pokazivač na funkciju za kreiranje konteksta.
- **set:** pokazivač na funkciju koja postavlja parametre procesiranja. Može se pozvati tek kada je pomoću init funkcije kreiran validan kontekst.
- **get:** pokazivač na funkciju koja vraća informacije o biblioteci i parametrima procesiranja.
- **process:** pokazivač na funkciju koja obrađuje dobijene odbirke i smešta ih u izlazni bafer.
- **analyze:** pokazivač na funkciju koja analizira ulazne odbirke sa ciljem da detektuje signal.
- **cleanup:** pokazivač na funkciju koja briše kontekst i oslobađa zauzete resurse.
- **reset:** pokazivač na funkciju koja resetuje kontekst.
- **audio_localization:** zastavica za uključivanje/isključivanje audio lokalizacije.
- **echo_cancel:** zastavica za uključivanje/isključivanje poništavanja eha.
- **noice_reduction:** zastavica za uključivanje/isključivanje smanjenja buke.
- **agc:** zastavica za uključivanje/isključivanje automatske regulacije pojačanja.
- **cbf:** zastavica za uključivanje/isključivanje prostornog filtriranja.
- **fil:** zastavica za uključivanje/isključivanje 100Hz filtriranja.

4.1.2 Funkcija za inicijalizaciju polja iz strukture stanja (seam_init)

Opis funkcionalnosti:

Inicijalizuje polja iz strukture stanja. U ovoj funkciji se otvara biblioteka sa algoritmima za obradu i dobijamo pokazivač na nju. Takođe se uz pomoć *dlsym* funkcije dobija adresa memorije gde je učitana određena funkcija. Na taj način dobijamo adrese funkcija koje smo učitali iz biblioteke i možemo da ih koristimo.

Zaglavlje funkcije:

```
static void seam_init(MSFilter *f);
```

Parametri funkcije:

- f: pokazivač na filter.

Povratna vrednost:

- funkcija nema povratnu vrednost.

4.1.3 Funkcija za oslobođanje polja iz strukture stanja (seam_uninit)

Opis funkcionalnosti:

Oslobađa polja iz strukture stanja.

Zaglavlje funkcije:

```
static void seam_uninit(MSFilter *f);
```

Parametri funkcije:

- f: pokazivač na filter.

Povratna vrednost:

- funkcija nema povratnu vrednost.

4.1.4 Funkcija za predobradu (seam_preprocess)

Opis funkcionalnosti:

Inicijalizacija struktura potrebnih za dalju obradu. Pribavljanje vrednosti pojedinih parametara iz prethodnih filtera u grafu. Pozivanje *init* funkcije koja vraća pokazivač na kontekst. U zavisnosti od parametara poziva se *set* funkcija za određenu obradu.

Zaglavlje funkcije:

```
static void seam_preprocess(MSFilter *f);
```

Parametri funkcije:

- f: pokazivač na filter.

Povratna vrednost:

- funkcija nema povratnu vrednost.

4.1.5 Funkcija za procesiranje (seam_process)

Opis funkcionalnosti:

U okviru ove funkcije obavlja se kompletna audio obrada. Na osnovu postavljenih parametara funkcijom *set*, bira se tip procesiranja i poziva se funkcija za obradu *process*. Funkcija za obradu zahteva “interleaved” ulazni bafer. On se kreira tako što se uzima po jedan odbirak sa mikrofona i sa zakašnjjenog ulaza i upisuje se u bafer. Nakon obrade podaci se prosleđuju na izlaz.

Zaglavljne funkcije:

```
static void seam_process(MSFilter *f);
```

Parametri funkcije:

- *f*: pokazivač na filter.

Povratna vrednost:

- funkcija nema povratnu vrednost.

4.1.6 Funkcija za krajnju obradu (seam_postprocess)

Opis funkcionalnosti:

Oslobađa zauzete resurse kao što su baferi za privremeno čuvanje odbiraka tokom obrade.

Zaglavljne funkcije:

```
static void seam_postprocess(MSFilter *f);
```

Parametri funkcije:

- *f*: pokazivač na filter.

Povratna vrednost:

- funkcija nema povratnu vrednost.

4.1.7 Funkcija za postavljanje zastavice za audio lokalizaciju (seam_set_audio_localization)

Opis funkcionalnosti:

Postavlja zastavicu koja uključuje ili isključuje audio lokalizaciju. Vrednost za zastavicu se prosleđuje iz korisničke sprege selekovanjem željene obrade.

Zaglavlj funkcije:

```
static int seam_set_audio_localization(MSFilter *f, void *arg);
```

Parametri funkcije:

- f: pokazivač na filter.
- arg: pokazivač na prosleđeni argument.

Povratna vrednost:

- povratna vrednost je nula.

U nastavku su navedene funkcije koje rade na isti način kao prethodno opisana sa razlikom da postavljaju za njih vezane zastavice:

- seam_set_noice_reduction(MSFilter *f, void *arg);
- seam_set_agc(MSFilter *f, void *arg);
- seam_set_beamforming(MSFilter *f, void *arg);
- seam_set_100Hz(MSFilter *f, void *arg);
- seam_set_echo_cancel(MSFilter *f, void *arg);

4.2 Korisnička sprega (UI) – proširenja

Korišćena su dva alata za proširenje korisničke sprege. Prvo su dodate komponente pomoću Glade okruženja, a zatim je iskorišćen Gtk+ API da se za komponente doda i funkcionalnost.

4.2.1 Struktura koja predstavlja mogućnost "libseamprocessing.so" biblioteke

```
typedef struct engineCpasO
{
    gboolean echo;
    gboolean audio_localization;
    gboolean noice_reduction;
    gboolean agc;
    gboolean cbf;
    gboolean fil;
} SeamCapabiltyies;
```

Polja:

- audio_localization: zastavica za uključivanje/isključivanje audio lokalizacije.
- echo_cancel: zastavica za uključivanje/isključivanje poništavanja eha.
- noice_reduction: zastavica za uključivanje/isključivanje smanjenja buke.

- **agc:** zastavica za uključivanje/isključivanje automatske regulacije pojačanja.
- **cbf:** zastavica za uključivanje/isključivanje prostornog filtriranja.
- **fil:** zastavica za uključivanje/isključivanje 100Hz filtriranja

4.2.2 Funkcija za pribavljanje statusa o izabranim opcijama (GetMedia2Caps)

Opis funkcionalnosti:

Pribavlja status o opcijama koje su izabrane u korisničkoj sprezi za *mediastreamer2* audio biblioteku.

Zaglavljne funkcije:

```
Media2Capabiltyies* GetMedia2Caps(void);
```

Parametri funkcije:

- Nema parametara

Povratna vrednost:

- Pokazivač na strukturu sa poljima koja predstavljaju izabране opcije.

4.2.3 Funkcija za pribavljanje statusa o izabranim opcijama (GetSeamCaps)

Opis funkcionalnosti:

Pribavalja status o opcijama koje su izabrane u korisničkoj sprezi za "libseamprocessing.so" audio biblioteku.

Zaglavljne funkcije:

```
SeamCapabiltyies* GetSeamCaps(void);
```

Parametri funkcije:

- Nema parametara

Povratna vrednost:

- Pokazivač na strukturu sa poljima koja predstavljaju izabране opcije.

4.2.4 Funkcija za inicijalizaciju parametara za biranje obrade (initCaps)

Opis funkcionalnosti:

Inicijalizuje parametre za biranje obrade. Inicijalno su sve opcije postavljene na *FALSE* u korisničkom interfejsu.

Zaglavljne funkcije:

```
void initCaps();
```

Parametri funkcije:

- Nema parametara

Povratna vrednost:

- Nema povratnu vrednost.

4.2.5 Funkcija za iscrtavanje izabranih opcija (linphone_gtk_init_m2_engine_caps)

Opis funkcionalnosti:

Popunjavanje opcija u korisničkom interfejsu koje su prethodno postavljene od strane *initCaps()* funkcije a odnose se na *mediastreamer2* biblioteku. Popunjavanje se takođe obavlja i pri svakoj promeni audio biblioteke.

Zaglavljne funkcije:

```
void linphone_gtk_init_m2_engine_caps(Media2Capabiltyies* media2Caps);
```

Parametri funkcije:

- *media2Caps*: pokazivač na strukturu sa parametrima.

Povratna vrednost:

- Nema povratnu vrednost.

4.2.1 Funkcija za iscrtavanje izabranih opcija (linphone_gtk_init_o_engine_caps)

Opis funkcionalnosti:

Popunjavanje opcija u korisničkoj sprezi koje su prethodno postavljene od strane *initCaps()* funkcije a odnose se na “libseamprocessing.so” biblioteku. Popunjavanje se takođe obavlja i pri svakoj promeni audio biblioteke.

Zaglavljne funkcije:

```
void linphone_gtk_init_o_engine_caps(SeamCapabiltyies* seam);
```

Parametri funkcije:

- *seam*: pokazivač na strukturu sa parametrima.

Povratna vrednost:

- Nema povratnu vrednost.

4.2.2 Funkcija za obradu zahteva za poništavanje eha (linphone_gtk_echo_cancelation_toggled)

Opis funkcionalnosti:

Funkcija koja se pokreće svaki put kada se izabere opcija za poništavanje eha u korisničkoj sprezi. Stanje izabrane opcije se čuva i prenosi nižim slojevima obrade koji će opciju proslediti do određenog filtera.

Zaglavljne funkcije:

- `void linphone_gtk_echo_cancelation_toggled(Gtkwidget *w);`

Parametri funkcije:

- `w`: pokazivač na *CheckBox* komponentu.

Povratna vrednost:

- Nema povratnu vrednost.

U nastavku su navedene funkcije koje rade na isti način kao prethodno opisana sa razlikom da opslužuju za njih vezane događaje iz korisničke sprege:

- `void linphone_gtk_agc_toggled(Gtkwidget *w);`
- `void linphone_gtk_noice_reduction_toggled(Gtkwidget *w);`
- `void linphone_gtk_audio_localization_toggled(Gtkwidget *w);`
- `void linphone_gtk_beamforming_toggled(Gtkwidget *w);`
- `void linphone_gtk_100Hz_filtering_toggled(Gtkwidget *w);`

4.2.3 Funkcija za postavljanje zastavice za izabranu obradu (EngineEchoStatus)

Opis funkcionalnosti:

Čuva stanje selektovane opcije u korisničkoj sprezi.

Zaglavljne funkcije:

- `void EngineEchoStatus(caps* caps, gboolean status);`

Parametri funkcije:

- `w`: pokazivač na *CheckBox* komponentu.

Povratna vrednost:

Nema povratnu vrednost

U nastavku su navedene funkcije koje rade na isti način kao prethodno opisana sa razlikom da čuvaju za njih vezane selektovane opcije:

- `void EngineAGCStatus(caps* caps, gint active, gboolean status);`

- void EngineAudioLocalizationStatus(caps* caps, gint active, gboolean status);
- void EngineNoiseReductionStatus(caps* caps, gint active, gboolean status);
- void EngineBeamformingStatus(caps* caps, gint active, gboolean status);
- void Engine100HzStatus(caps* caps, gint active, gboolean status);

4.3 “coreapi”- proširenja

U osnovnu spregu su dodata proširenja koja omogućavaju prosleđivanje parametra koji određuje koja će obrada u “seam” filteru biti odabrana. Takođe je dodatnim parametrima proširena funkcija iz *mediastreamer2* API-ja koja pokreće prenos podataka a ujedno kreira sve filtere i prosleđuje im parametre.

4.3.1 Funkcija za pokretanje prenosa podataka (audio_stream_start_full)

Opis funkcionalnosti:

Pokreće prenos podataka, kreira sve filtere, prosleđuje im neophodne parametre. U okviru ove funkcije se pokreće graf obrade uz pomoć “ticker” API-ja. Ova funkcija se nalazi u okviru *mediastreamer2* API-ja ali je opisana u ovom poglavlju jer predstavlja ključan deo koje obezbeđuje funkcionalnost. Neće biti opisani svi parametri nego samo oni kojima je ova funkcija proširena.

Zaglavljene funkcije:

- int audio_stream_start_full(AudioStream *stream, RtpProfile *profile, const char *remip, int remport, int rem_rtcp_port, int payload, int jitt_comp, const char *infile, const char *outfile, MSSndCard *playcard, MSSndCard *captcard, bool_t use_ec, bool_t use_seam_ec, bool_t use_seam_use_seam_audio_localization, bool_t use_seam_noice_reduction, bool_t use_seam_agc, bool_t use_seam_cbf, bool_t use_seam_fil);

Parametri funkcije:

- use_ec: zastavica koja obezbeđuje pokretanje algoritma za poništavanje eha iz *mediastreamer2* biblioteke.
- use_seam_ec: zastavica koja obezbeđuje pokretanje algoritma za poništavanje eha iz “libseamprocessing.so” biblioteke.
- use_seam_audio_localization: zastavica koja obezbeđuje pokretanje algoritma za audio lokalizaciju iz “libseamprocessing.so” biblioteke.
- use_seam_noice_reduction: zastavica koja obezbeđuje pokretanje algoritma za smanjenje buke iz “libseamprocessing.so” biblioteke.

- `use_seam_agc`: zastavica koja obezbeđuje pokretanje algoritma za automatsku regulaciju pojačanja iz “`libseamprocessing.so`” biblioteke.
- `use_seam_cbf`: zastavica koja obezbeđuje pokretanje algoritma za prostorno filtriranje iz “`libseamprocessing.so`” biblioteke.
- `use_seam_fil`: zastavica koja obezbeđuje pokretanje algoritma za 100Hz filtriranje iz “`libseamprocessing.so`” biblioteke.

Povratna vrednost:

- Povratna vrednos je nula.

4.3.2 Funkcija za postavljanje zastavice za biranje obrade (`linphone_core_enable_seam_echo_cancellation`)

Opis funkcionalnosti:

Omogućava ili onemogućava poništavanje eha.

Zaglavljne funkcije:

- `void linphone_core_enable_seam_echo_cancellation(LinphoneCore *lc, bool_t val);`

Parametri funkcije:

- `lc`: pokazivač na “`core`” struktura koja čuva zastavice.
- `val`: vrednost koja se prosleđuje iz korisničke sprege.

Povratna vrednost:

- Nema povratnu vrednost.

U nastavku su navedene funkcije koje rade na isti način kao prethodno opisana sa razlikom da čuvaju za njih vezane selektovane opcije:

- `void linphone_core_enable_seam_audio_localization(LinphoneCore *lc, bool_t val);`
- `void linphone_core_enable_seam_noice_reduction(LinphoneCore *lc, bool_t val);`
- `void linphone_core_enable_seam_agc(LinphoneCore *lc, bool_t val);`
- `void linphone_core_enable_seam_beamforming(LinphoneCore *lc, bool_t val);`
- `void linphone_core_enable_seam_100Hz_filtering(LinphoneCore *lc, bool_t val);`

4.3.3 Funkcija za očitavanje postavljene obrade (**linphone_core_echo_cancellation_enabled**)

Opis funkcionalnosti:

Izčitava iz strukture vrednost zastavice koja kontroliše opciju poništavanja eha.

Zaglavljne funkcije:

- `bool_t linphone_core_echo_cancellation_enabled(LinphoneCore *lc);`

Parametri funkcije:

- `lc`: pokazivač na “core” strukturu koja čuva zastavice.

Povratna vrednost:

- `true`: ako je poništavanje eha omogućeno
- `false`: ako poništavanje eha nije omogućeno

U nastavku su navedene funkcije koje rade na isti način kao prethodno opisana sa razlikom da vraćaju zastavicu vezanu za njima specifičnu obradu:

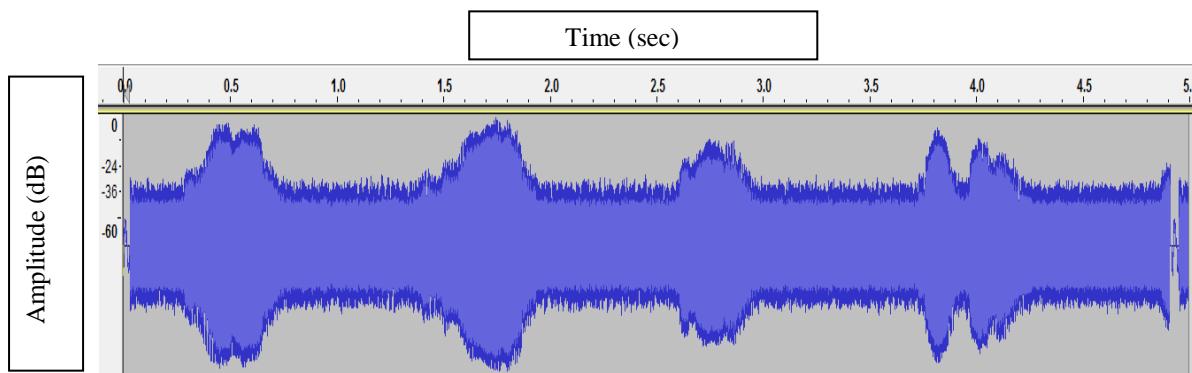
- `bool_t linphone_core_seam_audio_localization_enabled(LinphoneCore *lc);`
- `bool_t linphone_core_seam_noice_reduction_enabled(LinphoneCore *lc);`
- `bool_t linphone_core_seam_agc_enabled(LinphoneCore *lc);`
- `bool_t linphone_core_seam_beamforming_enabled(LinphoneCore *lc);`
- `bool_t linphone_core_seam_100Hz_filtering_enabled(LinphoneCore *lc);`

5. Rezultati

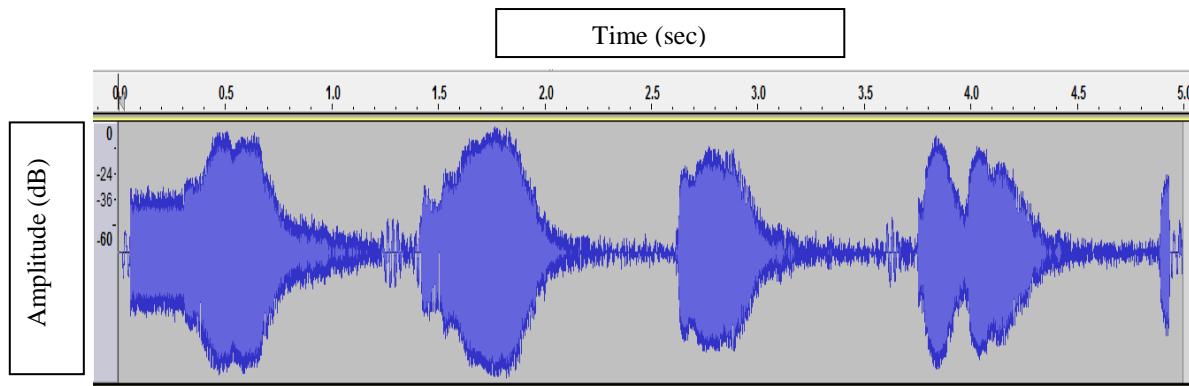
U cilju testiranja u tok podataka uključena su još dva filtera iz skupa filtera koji se nalaze u *mediastream2* biblioteci i to *MSFilePlay* i *MSFileRec*. Ideja je da se umesto uzimanja odbiraka sa mikrofona govor snimi u audio datoteku. Korišćenjem nekih od programa za obradu zvuka u signal iz datoteke se ubaci željeni efekat a zatim se datoteka uz pomoć gore navedenih filtera postavi kao ulaz umesto mikrofona, pusti kroz određenu obradu i potom snimi na prijemnoj strani.

5.1 Potiskivanje šuma

Potiskivanje šuma je testirano tako što je u testni signal preko aplikacije za obradu zvuka dodat beli šum. Signal sa šumom se koristeći prethodno navedene filtere dovodi na ulaz grafa obrade. U “seam” filteru se selektuje opcija za potiskivanje šuma. Očekuje se da je u izlaznom signalu šum u najvećoj meri potisnut.



Slika 5.1 – Signala sa šumom

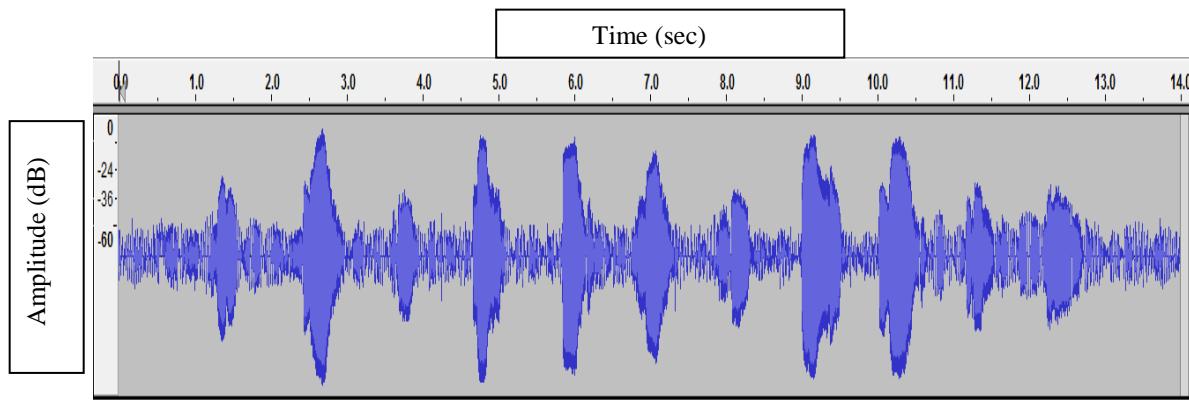


Slika 5.2 – Signal sa potisnutim šumom

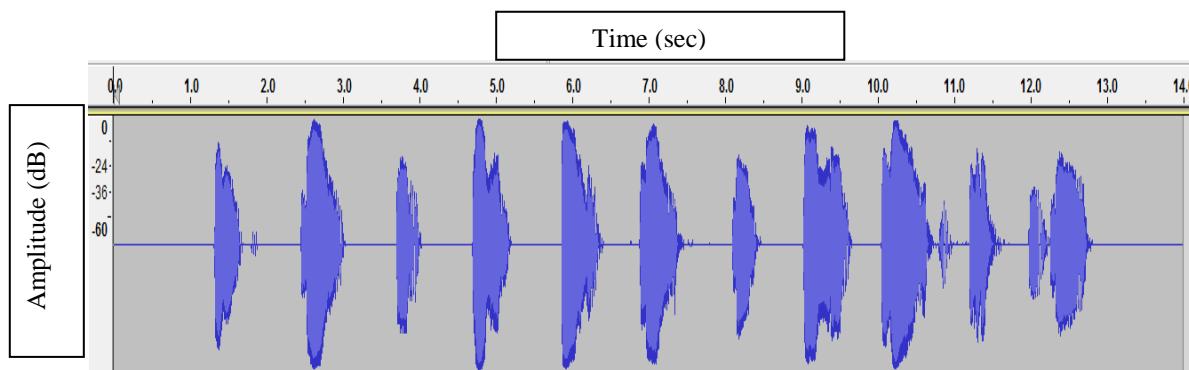
Sa dijagrama se može zaključiti da je šum u najvećem delu uklonjen te se može zaključiti da poništavanje eha funkcioniše.

5.2 Automatska kontrola jačine zvuka

Na ulaz grafa obrade se dovodi testni signal. U “seam” filteru se bira opcija za automatsku kontrolu jačine zvuka. Na izlazu se očekuje da su zvukovi srednje jačine na održani konstantnim dok oni niske jačine treba da su pojačani.



Slika 5.3 – Signal bez primenjene automatske kontrole jačine zvuka

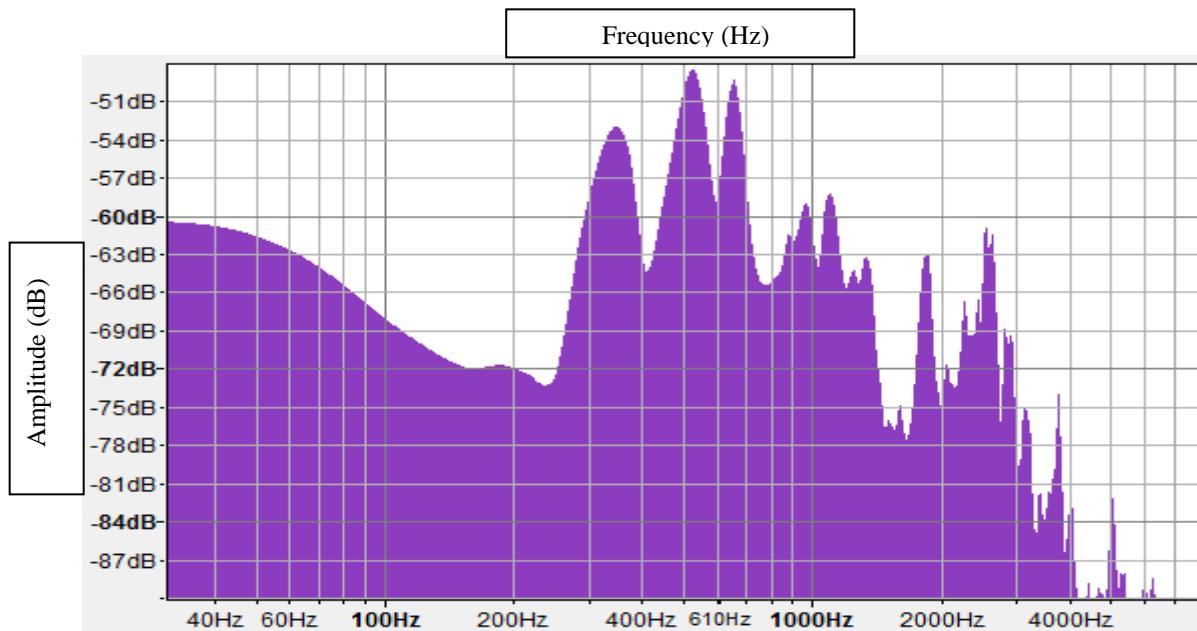


Slika 5.4 Signal sa primenjenom automatskom kontrolom jačine zvuka

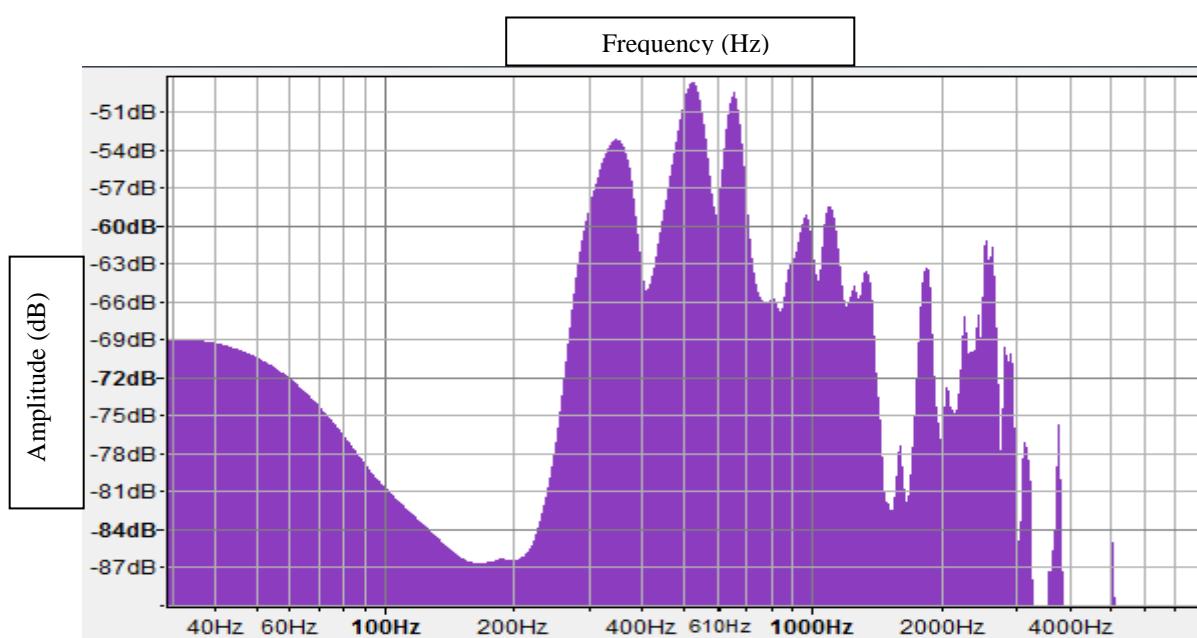
Sa dijagrama se može zaključiti da automatska kontrola jačine zvuka vrši kontrolu jačine tako što zvukove srednje jačine drži na istom nivou dok one jako niske pokušava da pojača.

5.3 100Hz filtriranje

Na ulaz sistema se dovodi signal u koji je aplikacijom za obradu zvuka unešen signal na 100Hz. U “seam” filteru se bira opcija za potiskivanje frekvencija ispod 100Hz. Na izlazu se očekuje signal u kome su frekvencije na 100Hz potisnute.



Slika 5.5 – Signal bez primjenjenog filtriranja na 100Hz

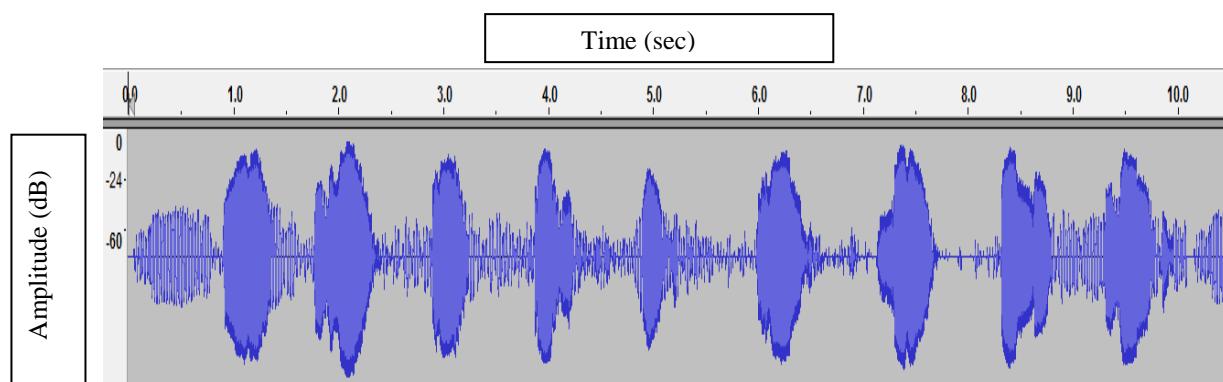


Slika 5.6 – Signal sa primjenjenim filtriranjem na 100Hz

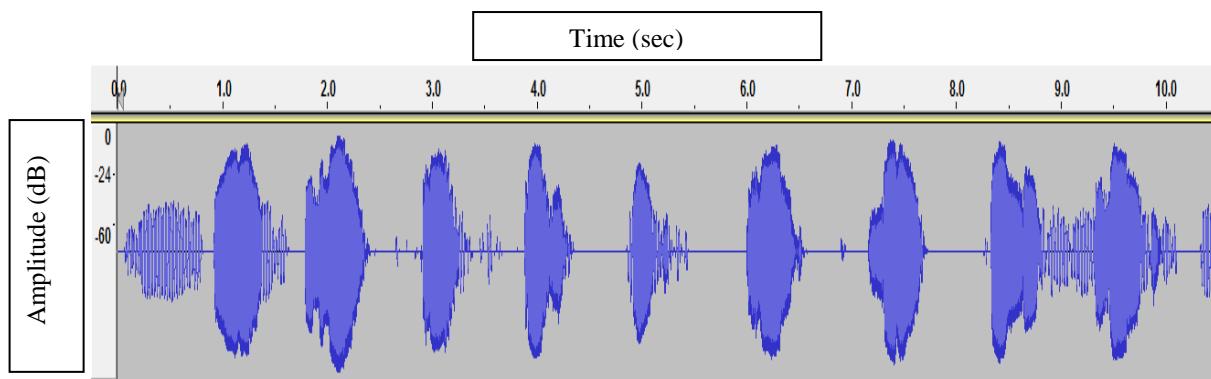
Sa spektralne karakteristike vidimo da primjenjeni filter potiskuje frekvencije ispod 100Hz čime se potvrđuje ispravnost rada ove komponente.

5.4 Prostorno filtriranje i audio lokalizacija

Pošto se u *Linphone* klijentskoj aplikaciji ne nalazi podrška za niz mikrofona testiranje audio lokalizacije i prostornog filtriranja nema smisla. Audio lokalizacija utiče na signal samo u kombinaciji sa prostornim filtriranjem. U sledećem testiranju samo pokazujemo da korišćenje opcije prostornog filtriranja i audio lokalizacije nema uticaja na izlazni signal.



Slika 5.7 – Testni signal

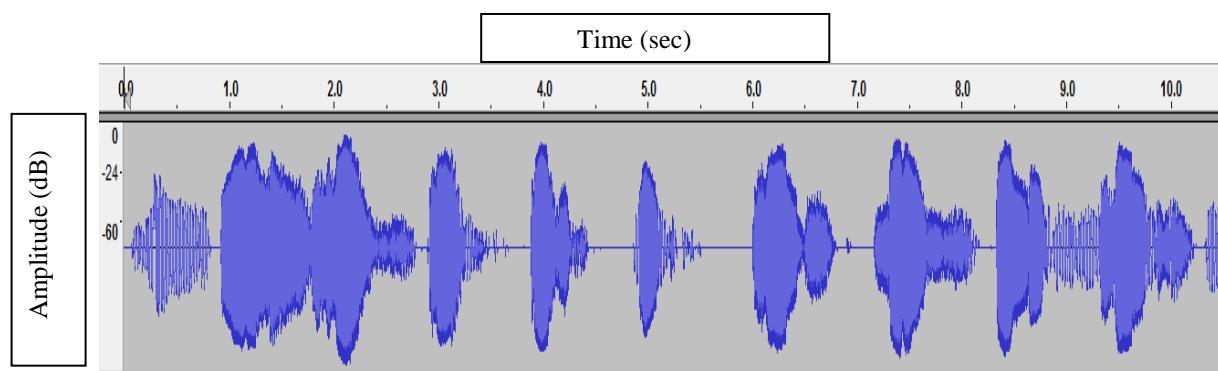


Slika 5.8 – Izlazni signal posle primjenjenog algoritma za prostorno filtriranje

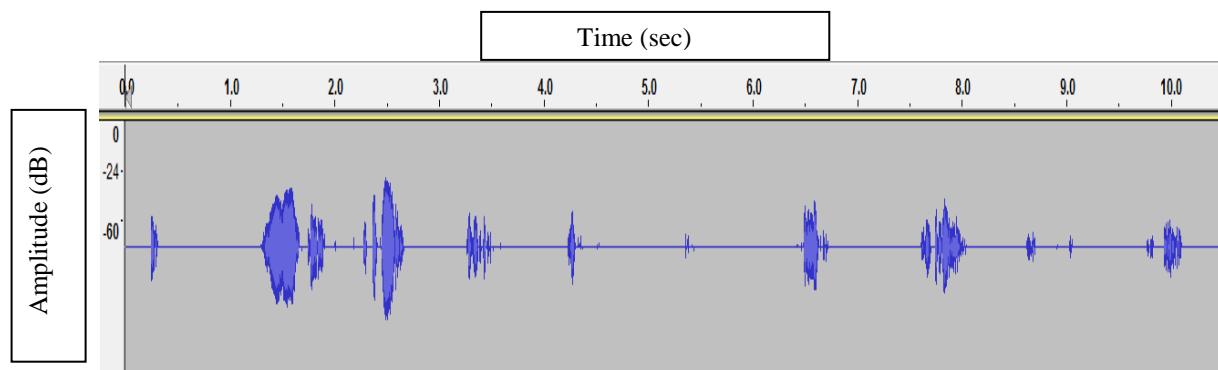
Sa karakteristikama vidimo da upotreba algoritama za prostorno filtriranje i audio lokalizaciju nemaju uticaja na izlazni signal.

5.5 Poništavanje eha

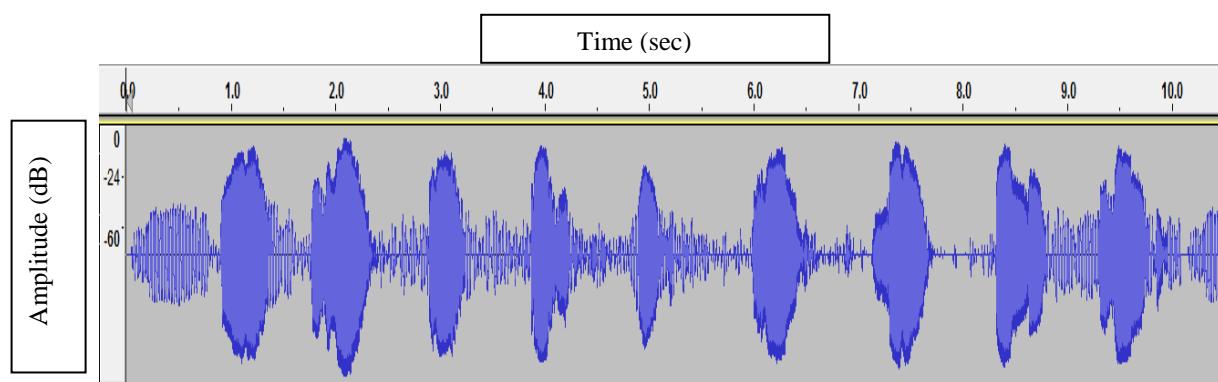
U testni signal je aplikacijom za obradu zvuka unešen echo. Taj signal je zatim doveden na ulaz sistema. U "seam" filteru se bira opcija za poništavanje eha. Na izlazu se očekuje signal sa poništenim ehom.



Slika 5.9 – Ulazni signal koji sadrži echo



Slika 5.10 – Echo signal koji treba ukloniti



Slika 5.11 – Izlazni signal sa uklonjenim ehom

Sa karakteristikama vidimo da je u izlaznom signalu poništen echo što je i očekivano. To dokazuje da je odgovarajuća obrada uključena i primenjena.

6. Zaključak

U ovom radu prikazano je rešenje integracije skupa funkcija za obradu audio podataka u postojeću klijentsku aplikaciju. Sam način rada aplikacije i neke njene funkcije su prilagođene novim funkcijama za obradu zvuka i na taj način je obezbeđena kompatibilnost. S obzirom da je zadržana stara funkcionalnost interakcija korisnika sa novim mogućnostima aplikacije je omogućena kroz proširenje korisničke sprege.

Aplikacija je testirana u realnim uslovima pa je na taj način verifikovan kvalitet prenosa i obrade audio podataka kako sa postojećim tako i sa dodatnim algoritmima za obradu.

Konferencijski pozivi predstavljaju deo većine VoIP aplikacija tako da i ta komponenta je moguća i poželjna. Takođe, proširenje može da bude biranje tipa obrade u toku trajanja sesije između dva ili više računara ukoliko se prethodno realizuju konferencijski pozivi. Aplikacija je prenosiva na različite operativne sisteme tako da uz male promene je i to moguće ostvariti. Većina današnjih računara sadrži umesto jednog niz mikrofona. To omogućava audio lokalizaciju koja je implementirana u okviru „*libseamprocessing.so*“ biblioteke ali nije iskorišćena zbog nedostatka „*Linphone*“ klijentske aplikacije. Ona prihvata signal samo jednog mikrofona, tako da implementacija podrške za niz mikrofona predstavlja još jednu moguću nadogradnju.

7. Literatura

- [1] Wikipedia: The Free Encyclopedia, <http://www.wikipedia.org>
- [2] Linphone: Free SIP VoIP Client, <http://www.linphone.org>
- [3] Henning Schulzrinne: “*The Session Initiation Protocol*”, Columbia University, New York, May 2001
- [4] Jean-Marc Valin: “*The Speex Codec Manual*”, Version 1.2 Beta 3, December 8, 2007
- [5] RT-RK: Computer Based Systems, <http://www.rt-rk.com/sea2m>