



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Игор Стефановић

Клијентска апликација за контролу паметне куће гласовним командама

ДИПЛОМСКИ РАД
- Основне академске студије -

Нови Сад, 2017



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:	Игор Стефановић		
Ментор, МН:	проф. др Иштван Пап		
Наслов рада, НР:	Клијентска апликација за контролу паметне куће гласовним командама		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2017		
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)	7/30/14/1/9		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	препознавање говора, кућна аутоматизација, наменски рачунар		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У раду је описана реализација модула за контролу периферних уређаја гласовним командама у паметним кућама. За препознавање говора искоришћен је Google Speech сервис. У раду је дат опис целокупне имплементације модула са свим логичким целинама које га чине. Функционалност имплементације је потврђена од стране постојећег система за аутоматизацију паметних кућа		
Датум прихваташа теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	доц. др Миодраг Ђукић	
	Члан:	проф. др Илија Башићевић	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:	Igor Stefanović	
Mentor, MN:	Ištván Papp, PhD	
Title, TI:	Application for voice control in smart home environment	
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:	2017	
Publisher, PB:	Author's reprint	
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6	
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/30/14/1/9	
Scientific field, SF:	Electrical Engineering	
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, S/KW:	voice recognition, home automation, embedded device	
UC		
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, N:		
Abstract, AB:	In this paper, one implementation of voice control module for smart home automation is proposed. The implemented module uses Google Speech as the speech-to-text service. Detailed explanation of the voice control module components and their implementation is presented. Functionality of the implementation is verified in the existing smart home automation system.	
Accepted by the Scientific Board on, ASB:		
Defended on, DE:		
Defended Board, DB:	President:	Miodrag Đukić, PhD
	Member:	Ilija Bašićević, PhD
	Member, Mentor:	Ištván Papp, PhD
		Menthor's sign

Zahvalnost

Ovim putem se posebno zahvaljujem svojoj porodici na pruženoj sveobuhvatnoj podršci tokom studiranja.

Zahvaljujem se i mentoru prof. dr Ištvanu Papu i stručnom saradniku dr Mariji Antić na savetima i pomoći tokom izrade ovog rada.



УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



SADRŽAJ

1.	Uvod	1
2.	Teorijske osnove.....	3
2.1	OBLO, sistem “pametne” kuće.....	3
2.1.1	Centralni uređaj	4
2.1.2	“Oblak” i njegovi servisi	5
2.1.3	Klijentske aplikacije	5
2.1.4	Komunikacija sa centralnim uređajem	5
2.2	Google Speech servis.....	6
2.3	POCO biblioteka.....	7
2.4	Prepoznavanje govora.....	7
2.4.1	Skriveni Markovski model	8
2.4.2	Viterbi algoritam	9
3.	Koncept rešenja	11
3.1	Arhitektura sistema	11
3.1.1	Modul za prepoznavanje ključne reči	12
3.1.2	Modul za snimanje komandi	12
3.1.3	Modul za konvertovanje i slanje datoteke	12
3.1.4	Modul za parsiranje odgovora	12
3.1.5	Modul za registraciju korisnika	13
3.1.6	Modul za formiranje i objavu sadržaja	14
4.	Programsko rešenje.....	15
4.1	Inicijalizacija sistema.....	15
4.2	Moduli	16

4.2.1	Modul za prepoznavanje ključne reči.....	16
4.2.2	Modul za snimanje komandi	17
4.2.3	Modul za konvertovanje i slanje datoteke	17
4.2.4	Modul za parsiranje odgovora.....	17
4.2.5	Modul za registraciju korisnika.....	17
4.2.6	Modul za formiranje i objavu sadržaja.....	18
5.	Rezultati.....	19
5.1	Teorijske osnove	19
5.2	Testiranje.....	20
6.	Zaključak	23
7.	Literatura	24

SPISAK SLIKA

Slika 2.1 Arihitektura OBLO sistema	4
Slika 2.2 Arhitektura centralnog uređaja	4
Slika 2.3 Google Speech servis JSON odgovor.....	6
Slika 2.4 Arhitektura POCO biblioteke	7
Slika 2.5 Primer snimljenog govora prikazanom u programu za izmenu zvuka	8
Slika 2.6 Primer Viterbi algoritma.....	9
Slika 3.1 Arhitektura projekta.....	11
Slika 4.1 Modul za prepoznavanje ključne reči.....	16
Slika 5.1 Mogući ishodi prepoznavanja ključne reči	20

SPISAK TABELA

Tabela 5.1 Rezultati testiranja kvaliteta..... 21

SKRAĆENICE

API	-Aplikaciona programska sprega (eng. Application Programming Interface)
MQTT	-Komunikacioni protokol za asinhronu razmenu poruka (eng. Message Queueing Telemetry Transport)
HTTP	-Mrežni protokol za prenos informacija (eng. HyperText Transfer Protocol)
FLAC	-Koder za audio kodovanje bez gubitaka (eng. Free Lossless Audio Codec)
PCM	-Impulsna kodna modulacija (eng. Pulse-code modulation)
LINEAR16	-Linearni PCM koder bez gubitaka
JSON	-Sintaksa za skladištenje i razmenu podataka (eng. JavaScript Object Notation)
ALSA	-Biblioteka za audio i MIDI u Linuks operativnom sistemu (eng. The Advanced Linux Sound Architecture)
MP3	-Audio format zapisa datoteke u kome je primenjena kompresija sa gubitkom (eng. MPEG-1 Audio Layer 3)
POCO	-Prenosive komponente (eng. The Portable Components)
WEB	-Svetska mreža (eng. World Wide Web)
DSP	-Digitalni procesor signala (eng. Digital signal processor)
TCP	-Transmisioni kontrolni protokol (eng. Transmission Control Protocol)
REST	-eng. Representational state transfer
gRPC	-eng. Remote procedure call razvijen od strane Google-a

1. Uvod

Povezivanje uređaja na Internet, a samim tim i mogućnost udaljene kontrole tako povezanih uređaja, sve više dobija na značaju u svetu tehnologija. Takođe, kontrola tih uređaja glasovnim komandama predstavlja jedan od najvećih inženjerskih izazova zadnjih godina.

Sistem za automatizaciju pametne kuće obuhvata različite periferne uređaje, koji se mogu podeliti u dve grupe: aktuatori i senzori. Aktuatori su uređaji čija svojstva korisnik može kontrolisati. Sa druge strane, vrednosti parametara senzora mogu se samo očitavati, u cilju nadgledanja sistema.. Periferni uređaji su povezani u jedinstven sistem “pametne” kuće preko centralnog uređaja (eng. *gateway*). Centralni uređaj omogućava komunikaciju između različitih perifernih uređaja, u slučajevima u kojima se želi ostvariti da promena stanja jednog od uređaja prouzrokuje promenu stanja nekog drugog perifernog uređaja, ili cele grupe. Sistemom pametne kuće se upravlja putem aplikacija sa mobilnog telefona ili iz Internet pretraživača. Međutim, poslednjih godina raste popularnost uređaja i aplikacija koji omogućavaju kontrolu glasom.

Cilj projekta izloženog u ovom radu je implementacija modula za kontrolu perifernih uređaja glasom, koji bi se izvršavao na namenskog sistemu. Modul bi predstavljaо deo centralnog uređaja ili bi mogao biti pokrenut na zasebnom namenskom uređaju zaduženom za obradu glasovnih komandi.

Rad je organizovan u 5 poglavlja. U prvom poglavlju opisane su teorijske osnove potrebne za razumevanje samog projekta. Tu spada kratak opis OBLO sistema kao jednog od rešenja sistema “pametnih” kuća, zatim način korišćenja Google Speech servisa koji predstavlja servis za prepoznavanje govora. Od značaja predstavlja i kratak opis matematičkih definicija vezanih za implementaciju sistema za prepoznavanje govora. U narednim poglavljima je objašnjen pristup rešenju projekta kao i programsko rešenje svih logičkih

celina koje ga čine. Zatim je prikazano testiranje tih logičkih celina od značaja za funkcionalnost celog sistema. U zaključku se govori o idejama za dalji razvoj i usavršavanje projekta, kao i preprekama prilikom realizacije.

2. Teorijske osnove

U ovom poglavlju biće ukratko opisane tehnologije korišćene za implementaciju osnovnih blokova modula za kontrolu glasom, kao tehnologije kojima se ostvaruje veza između tih blokova.

Prilikom razvoja sistema za kućnu automatizaciju, poseban inženjerski izazov predstavlja činjenica da treba objediniti uređaje koji koriste različite mrežne protokole. Glavnu ulogu u razvoju ovakvih sistema ima programska podrška. Sa jedne strane, potrebno je upravljati perifernim uređajima preko protokola podržanih od strane tih uređaja. Sa druge strane, potrebna je i interakcija sa korisnikom, kako bi korisnik dobio odgovarajuće informacije o svom sistemu, imao mogućnost udaljene kontrole i slično.

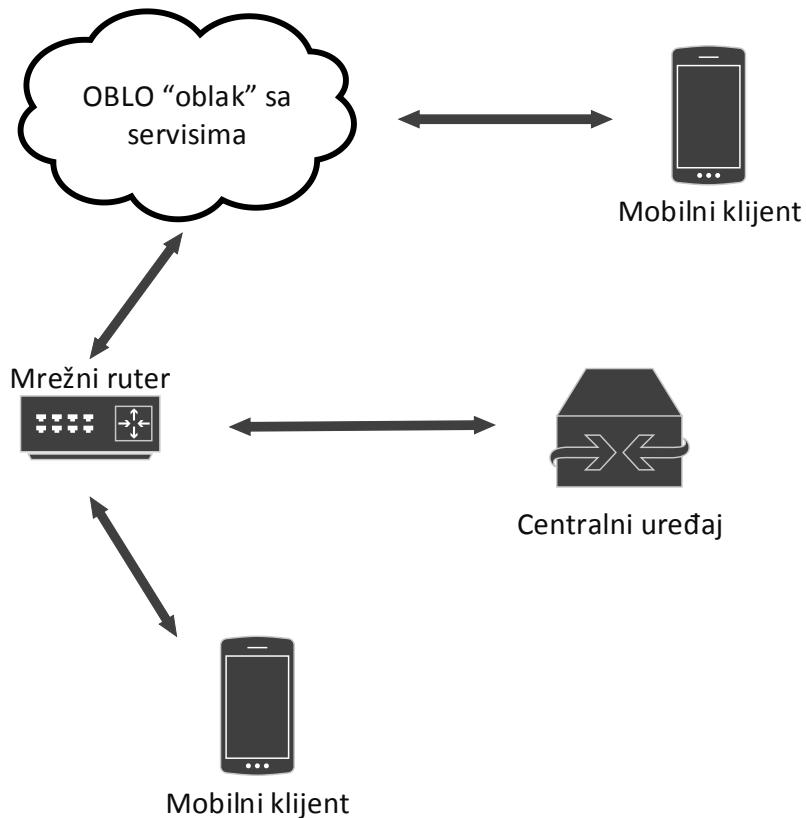
Sistem pametne kuće na kome se zasniva ovaj projekat je OBLO sistem kućne automatizacije razvijen od strane RT-RK instituta.

2.1 OBLO, sistem "pametne" kuće

OBLO sistem predstavlja sistem za automatizaciju kuće i sastoji se iz više nezavisnih celina koje komuniciraju međusobno posredstvom raznih komunikacionih protokola. Glavnu ulogu u sistemu ima centralni uređaj, čiji je zadatak da komande primljene od strane korisnika prosledi perifernim uređajima, kao i da informacije od značaja koje pristižu od perifernih uređaja prosledi korisniku.

Pored centralnog i perifernih uređaja, OBLO sistem čine i klijentske aplikacije. Njihov zadatak je da pruže korisniku uvid u trenutno stanje svih uređaja i da omoguće udaljenu kontrolu sa bilo kog mesta.

Spregu između korisnika i centralnog uređaja predstavlja "oblak" (eng. *cloud*). On omogućava sakupljanje podataka, pravljenje statistike, pravljenje istorije stanja uređaja, povezivanje naloga sa drugim klijentima poput Google-a i Amazon-a.

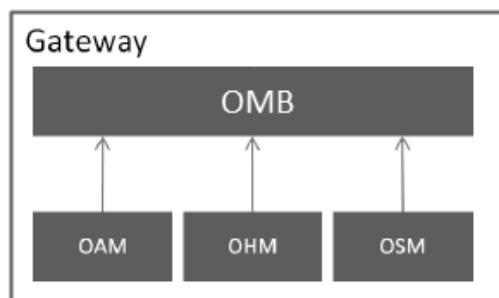


Slika 2.1 Arhitektura OBLO sistema

2.1.1 Centralni uređaj

Periferni uređaji su najčešće specificirani za samo određene funkcije, najpre zbog veće slobode raspoređivanja unutar prostora, ali i jednostavnije izrade samih uređaja.

Kako je svaki od ovih uređaja jedinstven po metodama koje podržava i po protokolu za komunikaciju koji koristi, njihovo apstrahovanje je jedan od glavnih problema. Da bi se prebrodilo problem razlika između protokola koje uređaji koriste, sama razmena podataka se oslanja na centralni uređaj, koji je u stanju da komunicira putem svih podržanih protokola. Neki od protokola koji služe za bežičnu komunikaciju između centralnog i perifernih uređaja su ZigBee [1] i Z-Wave [2]. Pored toga, zadatok centralnog uređaja je da grupiše periferne uređaje po njihovim funkcionalnostima.



Slika 2.2 Arhitektura centralnog uređaja

Kao što je prikazano na Slici 2.2, centralni uređaj čine 4 komponente. OMB (eng. *OBLO Message Broker*) predstavlja centar za distribuiranje MQTT poruka. U njemu se nalazi MQTT broker [3], zadužen je za autentikaciju korisnika, nadgledanje pristupanju temama i samo presljeđivanje poruka. OAM (eng. *OBLO Application Manager*) je menadžer WEB aplikacija. OHM (eng. *OBLO Home Manager*) je sistem zaslužan za upravljanje perifernim uređajima. OSM (eng. *OBLO System Manager*) je aplikacija koja se koristi za obezbeđivanje i održavanje različitih servisa specifičnih za hardersku platformu.

2.1.2 “Oblak” i njegovi servisi

U slučaju kada su i klijentska aplikacija i centralni uređaj u lokalnoj mreži, komunikacija između njih je uglavnom bežična i bez posrednika. Problem se javlja kada je klijentska aplikacija izvan lokalne mreže, jer tada mora postojati drugi način pristupa centralnom uređaju. Iz tog razloga, uvodi se “oblak” (eng. *cloud*), koji predstavlja poslužioca (eng. *server*) koji je konstantno povezan sa centralnim uređajem i dostupan na Internet-u. On vodi računa o slanju informacija između klijenata i odgovarajućih centralnih uređaja.

2.1.3 Klijentske aplikacije

Za kontrolu sistema, korisnik upotrebljava klijentsku aplikaciju koja se povezuje sa centralnim uređajem, posredstvom “oblaka” servisa ili preko lokalne mreže, radi dobavljanja informacija o sistemu. Osnovne informacije koje klijentska aplikacija dobavlja su lista dostupnih uređaja, kao i informacije o funkcionalnostima tih uređaja. Nakon toga, korisnik može upravljati uređajima individualno ili grupnim kontrolama.

2.1.4 Komunikacija sa centralnim uređajem

Komunikacija centralnog uređaja i „oblaka“, kao i centralnog uređaja i klijentskih aplikacija, ostvaruje se preko javno dostupnog MQTT protokola.

Sam protokol funkcioniše po principu pretplate i objave posredstvom komponente zvane **broker**. Klijenti se pretplaćuju ili objavljuju na određene “teme” (eng. *topic*). Informacije koje klijent objavljuje nalaze se u okviru dela poruke koji se naziva “sadržaj” (eng. *payload*). Sve poruke dolaze do brokera, koji ima zadatak da ih pravilno preusmeri onim klijentima koji su na temu te poruke pretplaćeni [3].

MQTT se oslanja na konstantnu TCP vezu, stoga klijentske aplikacije moraju imati mehanizam koji će stalno biti aktivan kako bi slanje bitnih informacija ka njima bilo moguće. Ovo može predstavljati problem kod mobilnih aplikacija, jer će taj mehanizam predstavljati stalno aktivovan pozadinski proces.

2.2 Google Speech servis

Google Speech servis omogućava korisnicima da konvertuju govor u tekst, korisćenjem modela neuronskih mreža uz veliku tačnost. Podržava sve uređaje koji imaju mogućnost slanja REST ili gRPC zahteva i pruža laku integraciju sa korisničkim aplikacijama. Speech servis vrši prepoznavanje jednom od tri metoda:

1. Sinhrono prepoznavanje govora vrši se tako što se šalju audio podaci ka Google Speech servisu, koji vrši prepoznavanje govora i odmah vraća odgovor u kome se nalazi prepoznata komanda. Zahtevi su ograničeni na dužinu audio podataka od 1 minuta.
2. Asinhrono prepoznavanje govora je metoda u kojoj se audio podaci šalju na Speech API i pokreće se *Long Running Operation*. Koristeći ovu operaciju, moguće je periodično preuzimanje rezultata prepoznavanja. Ova metoda se koristi sa zahteve sa dužinom trajanja do 80 minuta.
3. Prepoznavanje u realnom vremenu vrši prepoznavanje govora tako što se audio podaci kontinualno šalju kroz gRPC bidirekcioni tok. Zahtevi su takvi da obezbeđuju prepoznavanje u realnom vremenu u isto vreme dok korisnik govori.

U projektu, korišćen je pristup sinhronog prepoznavanja govora. U ovom slučaju, vrši se obrada komande iz lokalne audio datoteke u realnom vremenu. Maksimalno trajanje komande koja se na ovaj način može obraditi iznosi 1 minut. Protokol za razmenu datoteke zahteva da se izvrši HTTP *post* zahtev na Google Speech servis za prepoznavanje govora, pri čemu se enkodovana audio datoteka smešta u polju za podatke *post* zahteva. Audio podaci moraju biti u FLAC ili LINEAR16 formatu, zbog toga što su ovo formati koji govor enkoduju bez gubitaka, te se očekuje i bolji kvalitet prepoznavanja. Odgovor je u JSON formatu ima format prikazan na Slici 2.3.

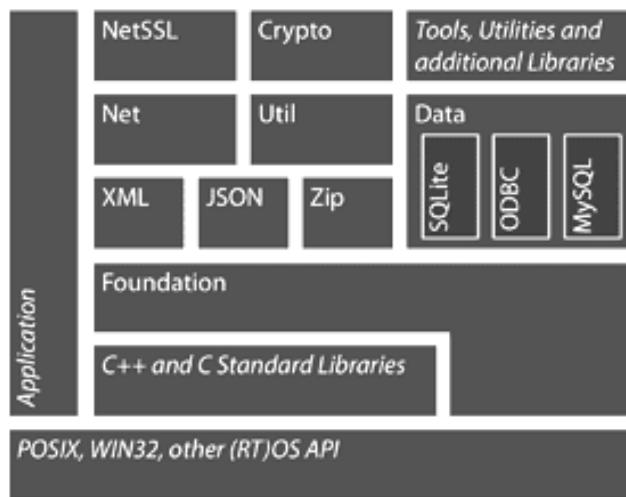
```
{
  "results": [
    {
      "alternatives": [
        {
          "transcript": "how old is the Brooklyn Bridge",
          "confidence": 0.98267895
        }
      ]
    }
  ]
}
```

Slika 2.3 Google Speech servis JSON odgovor

Polje *transcript* u JSON odgovoru koji je Google Speech servis vratio predstavlja rečenicu koja je prepoznata, a polje *confidence* predstavlja metriku kojom se meri kvalitet prepoznavanja. Vrednost polja *confidence* može biti u opsegu od 0 do 1.

2.3 POCO biblioteka

POCO je skup biblioteka [4] za potrebe mrežnih aplikacija pisanih u programskom jeziku C++. Odlikuje ih prenosivost i dostupnost za gotovo sve prisutne platforme (Windows, Linux, MIPS, itd), što je jedan od glavnih preduslova dobre programske podrške.

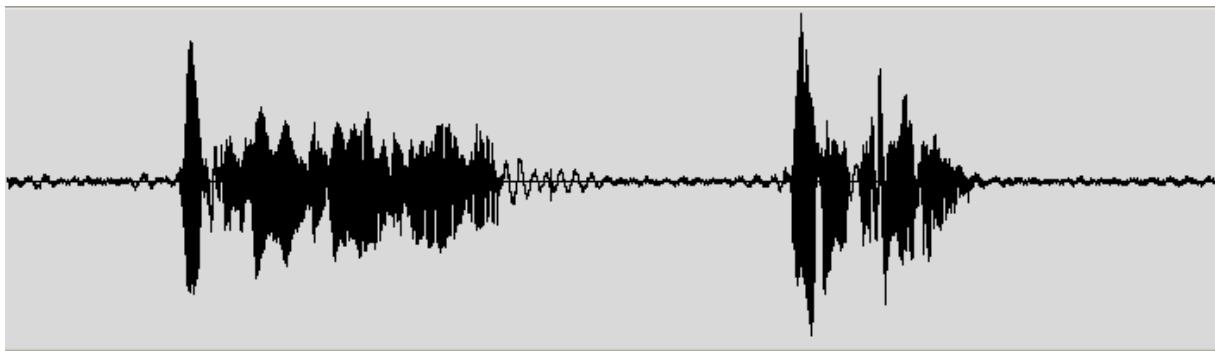


Slika 2.4 Arhitektura POCO biblioteke

Biblioteka omogućava lak i jednostavan rad sa XML datotekama, bazama podataka, JSON objektima kao i raznim programskim mehanizmima neophodnim za razvoj ovakvog tipa programske podrške. Implementacija POCO biblioteke predstavlja dobru praksu apstrahovanja operativnog sistema umetanjem jednog sloja koji leži između operativnog sistema i aplikacije. Njegovom upotreboru eleminiše se problem zavisnosti aplikacije od funkcija operativnog sistema za rad sa mrežama, datotekama, procesima i slično. Nije samo prenosivost prednost upotrebe POCO biblioteke, dostupne su i neke napredne strukture podataka, poput dinamičkih promenljivih i slično. Takođe, u velikoj meri je olakšan rad sa programskim nitima. Visoki nivo arhitekture prikazan je na Slici 2.3.

2.4 Prepoznavanje govora

Ljudski govor ima veoma specifične osobine zbog kojih ga je teško podvrgnuti obradi u računarstvu. Najčešći pristup koji se koristi jeste da se govor podeli na reči, i potom na foneme (eng. *phones*), od kojih se svaka posebno obrađuje određenim algoritmima, bez obzira na to što je govor dinamički proces bez jasno odvojenih delova.



Slika 2.5 Primer snimljenog govora prikazanom u programu za izmenu zvuka

Pod govorom se smatra kontinualna sekvenca stabilnih stanja kombinovanih sa dinamički promenljivim stanjima. U toj sekvenci stanja, mogu se definisati donekle slične klase fonema.

Prepoznavanje govora je proces konvertovanja govornih signala u niz reči, uz pomoć algoritma implementiranog kao računarski program. Većina sistema za prepoznavanje govora zasniva se na upotrebu skrivenog Markovskog modela, a kao arhitekturu koriste neuronske mreže.

2.4.1 Skriveni Markovski model

Procesi Markova su slučajni stohastički procesi bez memorije, što znači da “budućnost” ne zavisi od “prošlosti” već samo od “sadašnjosti”. [5] To je izraženo sledećom relacijom:

$$P(X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1}, \dots, X_{t_1} = x_1) = P(X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1})$$

Za razliku od klasičnih Markovskih modela, skriveni Markovski modeli podrazumevaju da posmatrani podaci ne predstavljaju sama stanja modela, vec se smatra da su generisani od strane skrivenih stanja [6].

Definicija: Uređeni par stohastičkih procesa $\{X, Y\}$, gde je $X = \{X_k\}_{k \in \mathbb{N}}$ i $Y = \{Y_k\}_{k \in \mathbb{N}}$, naziva se skriveni lanac Markova ako je X lanac Markova koji se ne može direktno opažati, a $Y_k = f(X_k, \omega_k)$, gde je f Borelova funkcija i $\{\omega_k\}_{k \in \mathbb{N}}$ niz nezavisnih jednako raspodeljenih slučajnih promenljivih koje su nezavisne i od X . Proces Y se zove proces opažanja [6].

Generalno, forma skrivenih Markovskih modela ima sledeća obeležja:

1. $\{X\} = \{X_1, X_2, \dots\}$ je lanac Markova koji ne možemo direktno opažati, takozvani niz signala,
2. $\{Y\} = \{Y_1, Y_2, \dots\}$ je niz opažanja,
3. N je dimenzija skupa stanja lanca Markova,
4. M je dimenzija skupa stanja niza opažanja,
5. $S_X = \{s_1, \dots, s_N\}$ je skup stanja lanca Markova

6. $O_Y = \{o_1, \dots, o_M\}$ je skup stanja niza opažanja
7. Matrica verovatnoće prelaza $\Pi = [\pi_{ij}]_{i,j=1,\dots,N}$ definisana sa:
$$\pi_{ij} = P(X_{k+1} = s_j | X_k = s_i), i, j = 1, \dots, N$$
8. Uslovna raspodela verovatnoća simbola opažanja $B_{M \times N} = [b_{ij}]_{i=1,\dots,N, j=1,\dots,M}$.

Funkcija verovatnoće za svako stanje i je data sa:

$$b_{ij} = P(Y_k = o_j | X_k = s_i)$$

9. Raspodela početnog stanja: $A_{N \times 1} = a_i, i = 1, \dots, N$

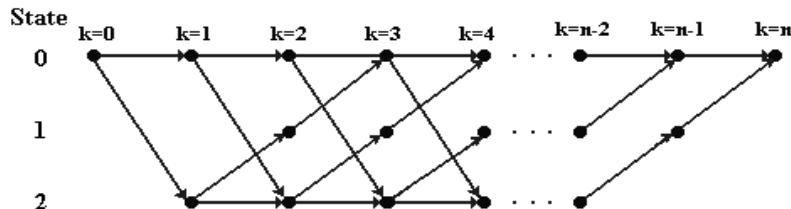
Definicija samog modela:

$$\lambda = (A, B, \Pi)$$

Skriveni Markovski modeli su poznati po primeni u temporalnim obrascima prepoznavanja kao što je govor, rukopis, gestikulacija i bioinformatika.

2.4.2 Viterbi algoritam

Viterbi algoritam predstavlja algoritam dinamičkog programiranja koji služi za pronalaženje najverovatnije sekvence skrivenih stanja (Viterbijevog puta), koji proizilazi iz sekvene posmatranih događaja, posebno u kontekstu skrivenih Markovskih modela [7][8].



Slika 2.6 Primer Viterbi algoritma

Prepostavimo da nam je dat skriveni Markovski model sa sledećim vrednostima: skup stanja S , raspodela početnog stanja A i uslovna raspodela verovatnoća simbola opažanja B . Recimo da posmatramo izlaz y_1, \dots, y_T , najverovatnija sekvenca stanja x_1, \dots, x_T koja proizvodi opažanje je data rekurentnim relacijama:

$$V_{1,k} = P(y_1 | k) * a_k$$

$$V_{t,k} = \max_{x \in S} (P(y_k | k) * b_{x,k} * V_{t-1,x})$$

Ovde je $V_{t,k}$ verovatnoća najverovatnije sekvence stanja $P(x_1, \dots, x_T, y_1, \dots, y_T)$ odgovorne za prvih t opažanja koje imaju k za svoje konačno stanje. Viterbijev put se može rekonstruisati čuvanjem pokazivača koji pamte koje stanje x je korišćeno u drugoj jednačini.

Neka je $Ptr(k, t)$ funkcija koja vraća vrednost x korišćenu da se izračuna $V_{t,k}$ ako je $t > 1$, ili k ako je $t = 1$. Onda važi:

$$x_T = \arg \max_{x \in S} (V_{T,x})$$

$$x_{t-1} = Ptr(x_t, t)$$

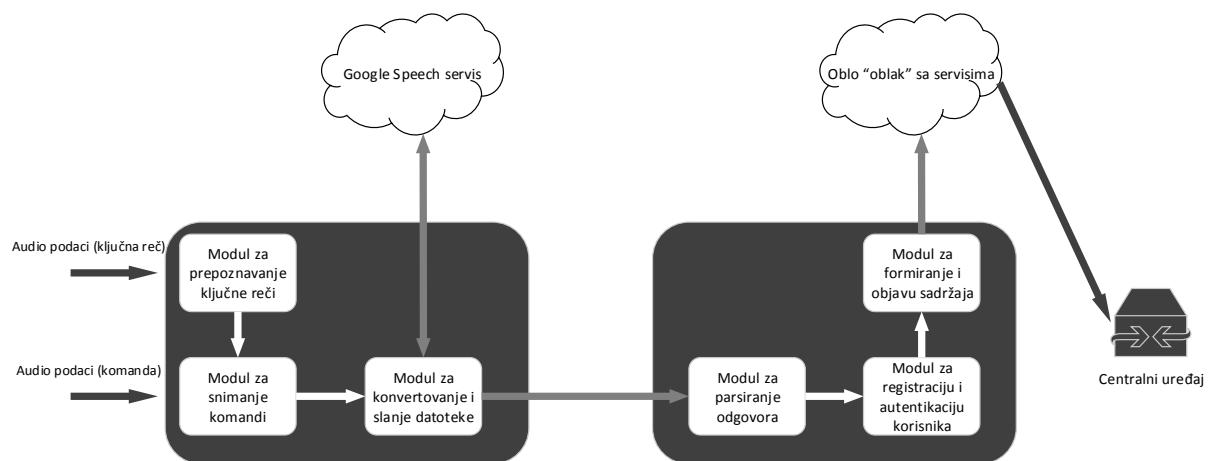
Ovde, $\arg\max$ predstavlja skup vrednosti x za koje funkcija $V_{T,x}$ ostvaruje maksimum.

3. Koncept rešenja

Sam projekat se zasniva na razvoju modula sa funkcionalnošću prepoznavanja govora, koji se može koristiti za kontrolu pametne kuće. Implementirani modul bi trebalo da unapredi rad postojećeg centralnog uređaja OBLO sistema, ili da posluži kao osnova za razvoj perifernog uređaja za kontrolu OBLO sistema glasom.

3.1 Arhitektura sistema

Sistem je podeljen u module. Modularna arhitektura olakšava dalji razvoj funkcionalnosti, i omogućava lako dodavanje novih modula. Takođe, moduli bi se mogli pokretati u okviru različitih procesa, ili na različitim računarima koji ostvaruju međusobnu komunikaciju. Komunikacija između modula se svodi na jednostavnu razmenu zastavica (eng. *flag*), osim posle prepoznavanje od strane Google Speech servisa kada se ceo JSON objekat koji je pristigao od Google Speech servisa prenese do modula za njegovo parsiranje i sređivanje.



Slika 3.1 Arhitektura projekta

Arhitekturu sistema čine modul za prepoznavanje ključne reči, modul za snimanje komandi korisnika, modul za konvertovanje MP3 datoteke i slanje te datoteke, modul za parsiranje odgovora, modul za autentikaciju korisnika, kao i modul za formiranje odgovarajućeg sadržaja njegovu objavu na određeni centralni uređaj.

3.1.1 Modul za prepoznavanje ključne reči

Na osnovu postojećih primera implementacije uređaja za kontrolu glasom, kako bi se korisniku omogućilo da glasom može pokrenuti sam sistem, uveden je modul za prepoznavanje ključne reči. Zadatak ovog modula je konstantno slušanje zvuka iz okruženja. U trenutku kada korisnik izgovori odgovarajuću ključnu reč, modul za prepoznavanje ključne reči treba da pokrene modul za snimanje komandi, a samim tim i ostatak sistema.

3.1.2 Modul za snimanje komandi

Nakon što je korisnik izgovorio ključnu reč, pokreće se modul za snimanje komandi. U tom trenutku, od korisnika se očekuje da izgovori željenu komandu. Uloga modula je da odgovarajuću komandu snimi u audio datoteku. Poseban problem predstavlja određivanje vremenskog okvira u kom je neophodno snimati komandu. Naime, komande mogu biti promenljive dužine, te nije pogodno koristiti konstantnu vrednost za vremenski interval u kome se vrši snimanje. Stoga je u okviru ovog modula implementiran i modul za prepoznavanje tišine, tj. modul za prepoznavanje kraja komande. Na taj način je smanjeno vreme odziva samog sistema u slučaju kratkih komandi, a izbegnut je problem odsecanja krajeva dužih komandi.

3.1.3 Modul za konvertovanje i slanje datoteke

Kako bi prepoznavanje od strane Google Speech servisa bilo najbolje moguće, sam API predlaže korišćenje nekog od kodera bez gubitaka, među kojima je i FLAC koder. Modul za konvertovanje i slanje datoteke, snimljenu datoteku koja sadrži komandu konvertuje u FLAC datoteku i, poštujući Google Speech API, šalje je servisu za prepoznavanje govora.

3.1.4 Modul za parsiranje odgovora

Odgovor dobijen od strane servisa za prepoznavanje govora je u JSON formatu i sadrži, pored prepoznatog govora, još neke moguće varijacije onoga što je prepoznato, te je potrebno njegovo parsiranje. Parsiranje odgovora se svodi na odbacivanje nepotrebnih karaktera i varijacija, tj. zadatak je svesti odgovor na jedan niz karaktera koji bi bio pogodan za dalju obradu.

3.1.5 Modul za registraciju korisnika

Sastavni deo MQTT protokola su identifikator korisnika, njegovo ime i lozinka. Za razliku od imena, identifikator je specifičan za korisnika i mora biti jedinstven MQTT brokeru. Kao i identifikator, i ime i lozinka se šalju pri uspostavi veze [3].

Procedura registrovanja novog korisnika se sastoji iz 8 koraka:

1. Prvo se zahteva uspostava inicijalne veze sa brokerom. Tom prilikom, korisnik se povezuje upotrebom imena definisanog isključivo za namenu registracije novih klijenata.
2. Nakon uspešno obavljenе uspostave veze za registraciju, klijent šalje zahtev za registraciju brokeru.
3. Kao potvrdu prihvaćene registracije, broker odgovara i dostavlja registracioni broj.
4. Nakon uspešno uspostavljenе veze, broker inicira mehanizam kojim proverava korisnika. U zahtevu, broker prosleđuje nasumično odabranu sekvencu brojeva, koji će korisnik u narednom koraku iskoristiti kako bi dokazao pravo korišćenje sistema.
5. Kako bi prihvatio novog korisnika, broker očekuje odgovor koji sadrži broj šifrovan pomoću standarda za napredno kodovanje (AES). Samo šifrovanje treba da bude obavljen na osnovu sledećih parametar:
 - Ključ: SHA1 [10] kod dobijen na osnovu posebno definisanog tajnog ključa
 - Format zapisa: niz heksadecimalnih brojeva
 - Dužina ključa: 128 bita (16 bajta)
 - Način kodovanja: ECB [11] i
 - Način popunjavanja: PKCS5 [12].
6. Rezultat šifrovanja je u binarnom obliku, tako da je potrebno taj rezultat prebaciti u tekstualni i poslati kroz poruku odgovora. Za to je korišćen Base64 [13] algoritam.
7. Ukoliko se broj iz odgovora poklopi, odnosno ukoliko je klijent upoznat sa tajnim ključem i procedurom kodovanja, broker će nastaviti sa procesom registracije. Naredni zahtev se odnosi na promenu korisničkog imena i lozinke.
8. Na klijentu je da odgovori i prosledi registracioni broj koji mu je dostavljen na početku procedure. Na taj način broker poseduje potrebne informacije o klijentu koji je započeo proces registracije. Nakon ovog koraka, klijentu je dozvoljeno da uspostavi regularnu vezu sa brokerom.

Tajni ključ može biti:

1. Predefinisan – dolazi iz brokera i može se naknadno promeniti

-
2. Periodično obnavljan – automatski se osvežava i može se dobiti uz pomoć korisnika koji je već registrovan
 3. Jednokratno generisan – generiše se po zahtevu nekog od registrovanih korisnika i važi za jednu registraciju

Kako bi se korisnici razlikovali, potrebno je pozvati niz funkcija koje izvršavaju svaki od koraka za registraciju korisnika.

3.1.6 Modul za formiranje i objavu sadržaja

Kako bi se komanda koju je korisnik zadao glasom izvršila, potrebno je pripremiti sadržaj MQTT poruke u skladu sa servisima koje ciljani uređaj podržava, a prema MQTT protokolu koji definiše način komunikacije između klijentskih aplikacija i centralnog uređaja. Formiranje sadržaja predstavlja glavni zadatak ovog modula. Pošto se sadržaj formira, vrši se njegova objava na odgovarajući centralni uređaj, koji tu komandu dalje prosleđuje odgovarajućem perifernom uređaju.

Nakon objave sadržaja, do modula za prepoznavanje ključne reči dolazi signal da on može nastaviti sa radom.

4. Programsко rešenje

Pretpostavljeno je da će se projekat izvršavati na namenskom računaru ograničenih resursa, pa je kod pisan u C i C++ programskim jezicima. Logički, projekat je podeljen u dve velike celine, takve da se jedna celina u potpunosti fokusira na čekanje i prepoznavanje ključne reči, snimanje zvuka, njegovo kodovanje u odgovarajući format i slanje na odgovarajući poslužilac. Druga celina se fokusira na pripremanje i objavljivanje sadržaja i sve ostalo što je potrebno da objava bude uspešna.

4.1 Inicijalizacija sistema

Pri startovanju, sistem inicijalizuje potrebne promenljive, konstante i određene konfiguracione parametre.

Modul za prepoznavanje ključne reči zauzima potrebne bafere, određuje ključnu reč na koju će se sistem pokretati i konfiguracione parametre potrebne za biblioteku koja vrši prepoznavanje ključne reči.

Modul za snimanje komande inicijalizuje rukovalac zvuka. Pozivaju se funkcije za otvaranje konekcije ka zvučnoj kartici. Zatim se zauzimaju resursi za smeštanje konfiguracionih parametara hardvera i vrši podešavanje tih konfiguracionih parametara na podrazumevane vrednosti. U konfiguracione parametre spadaju: način rada same zvučne kartice, format odbiraka i frekvencija njihovog odabiranja, veličina blokova, itd.

Druga celina u inicijalizaciji ima za zadatak da učita moguće komande za podržani jezik. Tom prilikom, parsira se XML datoteka, koja sadrži moguće kombinacije komandi sa vrednostima koje trebaju biti podešene da bi periferni uređaj izvršio komandu. Posle parsiranja, ti podaci se smeštaju se u mape vezane za tip uređaja. Naime, periferni uređaji podržavaju različite servise, te se i moguće komande za svaki servis nalaze u okviru odgovarajuće mape. U mapi, ključ predstavlja tekst podržane komande, dok je vrednost u

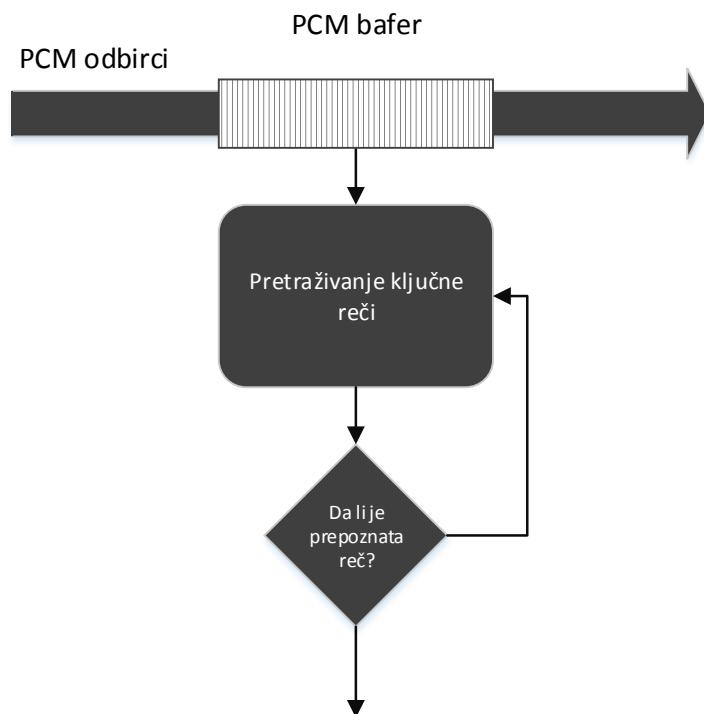
mapi ekvivalentna vrednosti odgovarajućeg parametra koji treba postaviti na perifernom uređaju.

4.2 Moduli

4.2.1 Modul za prepoznavanje ključne reči

Kako bi sistem konstantno slušao i čekao da se aktivira na odgovarajuću ključnu reč, modul za prepoznavanje ključne reči se izvršava kao pozadinski proces. Zvuk koji dolazi iz okruženja smešta se u PCM bafer koji je implementiran tako da se u njega upisuju podaci u blokovima. Novi blokovi podataka se upisuju na početak samog bafera, dok se stari sa kraja brišu i svi ostali se pomere za veličinu jednog bloka ka kraju. Ovim se postiglo to da bafer konstantno u sebi sadrži tačno dve sekunde trenutnih podataka, bez većeg kašnjenja i propuštanja podataka. Nad ovim baferom se izvršava obrada, odnosno, prepoznavanje ključne reči pomoću biblioteke Pocketsphinx [6]. Pocketsphinx je biblioteka za prepoznavanje govora projektovana za namenske računare. Pocketsphinx ne zahteva vezu sa Internetom da bi radio, i za potrebe ovog projekta, podešen je da radi u režimu prepoznavanja ključne reči. Pocketsphinx biblioteka se bazira na Viterbi algoritmu, spomenutom u Odeljku 2.4.2.

Nakon što Pocketsphinx biblioteka prepozna ključnu reč, ovaj modul prestaje sa radom dok do njega ne dođe signal za nastavak rada. Sama pozadinska nit ne prestaje sa radom, jer i dalje puni PCM bafer podacima, kako bi uvek sadržao trenutne podatke, ali nema obrade nad tim podacima.



Slika 4.1 Modul za prepoznavanje ključne reči

4.2.2 Modul za snimanje komandi

Snimanje komandi se pokreće nakon prepoznate reči. Snimanje se obavlja pomoću ALSA biblioteke koja predstavlja spregu ka rukovaocima za zvuk. Snimljeni zvuk se smešta u prethodno napravljenu MP3 datoteku. U okviru ovog modula je implementiran i modul za prepoznavanje kraja komande.

U toku samog snimanja komande, računa se i snaga svakog pojedinačnog bloka odbiraka formulom:

$$P = \frac{\sum_{i=0}^N x(i)^2}{N}$$

gde N predstavlja veličinu bloka, $x(i)$ vrednost signala u tom odbirku, a P snagu bloka odbiraka. Kako bi funkcionalnost bila smislena, uzet je broj blokova koji predstavljaju ~ 0.3 sekunde audio podataka i, ako je snaga u svakom od tih blokova pojedinačno bila ispod unapred zadatog praga, snimanje se završava.

4.2.3 Modul za konvertovanje i slanje datoteke

Kako bi se zadovoljio API Google Speech servisa, potrebno je prethodno snimljenu datoteku konvertovati u neki od kodera bez gubitaka. U projektu je korišćen FLAC koder. Za konvertovanje se koristi FLAC biblioteka.

FLAC datoteka se zatim pomoću CURL biblioteke šalje Google Speech servisu na prepoznavanje komande. Odgovorom koji pristiže rukuje funkcija povratnog poziva (eng. *callback*) koji podatke HTTP odgovora smešta u odgovarajući niz karaktera.

4.2.4 Modul za parsiranje odgovora

Kako je format poruke koju je poslao Google Speech servise u JSON formatu, potrebno je njegovo parsiranje i svođenje na prost niz karaktera koji je najjednostavniji za dalju obradu. U obzir se uzima samo *transcript* polje sa najvećom vrednošću u polju *confidence* jer se upravo u tom polju nalazi najverniji prikaz onoga što je korisnik izgovorio.

Ova funkcionalnost je ostvarena upotrebom POCO biblioteke.

4.2.5 Modul za registraciju korisnika

Pri prvom pokretanju sistema, ovaj modul izdaje sekvencu komandi za registraciju korisnika koja se odigrava u 8 koraka opisanih u Odeljku 3.1.5. Svaki sledeći put, modul za registraciju korisnika samo podešava parametre tako da registrovani korisnik bude prepoznat na centralnom uređaju. Naime, centralni uređaj vrši proveru korisnika kako bi mu se dozvolila određena prava u celokupnom sistemu pametne kuće, tj. mogućnost kontrole perifernih uređaja.

4.2.6 Modul za formiranje i objavu sadržaja

Formiranje sadržaja i njegova objava se svodi na poređenje odgovarajućih delova izrečene i prepoznate komande sa ključevima iz prethodno inicijalizovanih mapa. Ako dodje do poklapanja, podešavaju se odgovarajući parametri u cilju pripreme sadržaja za njegovu objavu. Uzimamo u obzir da je komanda izgovorena tako da sadrži željenu akciju, ime uređaja nad kojim želi da se zadata akcija izvrši, kao i neke dodatne parametre, ako su potrebni za uspešno izvršavanje te akcije.

Podešeni parametri po MQTT protokolu formiraju sadržaj i zatim se sadržaj objavljuje centralnom uređaju koji po tim parametrima kontroliše određeni uređaj.

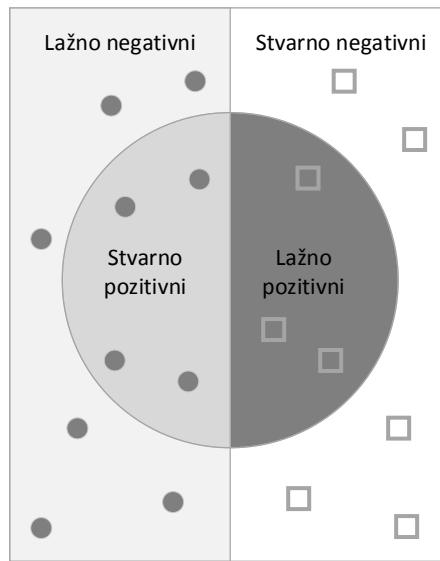
5. Rezultati

Tačke projekta od važnosti za testiranje su moduli koji mogu uneti najveće greške i najveće kašnjenje. Jedna od tih tačaka predstavlja modul za prepoznavanje ključne reči, jer u njemu funkcionalnost Pocketsphinx biblioteke troši najviše procesorke snage kao i najviše radne memorije. Takođe, potrebno je i testirati sposobnost tačnog prepoznavanja Pocketsphinx biblioteke, odnosno, njegovu preciznost, tačnost i odziv. Deo projekta koji je takođe od značaja i koji unosi najviše kašnjenja jeste Google Speech servis, čije prepoznavanje predstavlja deo na koji se ne može uticati direktno. Potrebno je dokazati da je kašnjenje koje servis unosi ne utiče mnogo na odziv sistema.

5.1 Teorijske osnove

Meru kvaliteta prepoznavanja govora, odnosno, u slučaju ovog projekta, prepoznavanje ključne reči, predstavlja tačnost (eng. *accuracy*), odziv (eng. *recall*) i preciznost (eng. *precision*) [14]. Pre svega, potrebno je definisati sve moguće ishode prilikom testiranja:

1. Stvarno pozitivni, SP (eng. *true positives*, *TP*) predstavlja broj puta kada je korisnik izgovorio ključnu reč, i modul je uspešno prepoznao
2. Stvarno negativni, SN (eng. *true negatives*, *TN*) predstavlja broj puta kada je korisnik izgovorio bilo koju drugu reč, i modul je tačno pretpostavio da ključna reč nije izgovorena
3. Lažno pozitivni, LP (eng. *false positives*, *FP*) predstavlja broj puta kada je korisnik izgovorio bilo koju drugu reč koja je pogrešno prepoznata kao ključna reč
4. Lažno negativni, LN (eng. *false negatives*, *FN*) predstavlja broj puta kada je korisnik izgovorio ključnu reč, a modul je nije prepoznao



Slika 5.1 Mogući ishodi prepoznavanja ključne reči

Preciznost koja ocenjuje tačnost klasifikacije, odnosno, koliki je procenat primera za testiranje je ispravno klasifikovan. Izračunava se po formuli:

$$\text{Preciznost} = \frac{SP}{SP + LP}$$

Odziv (pokrivanje) koji ocenjuje koliko je model uspešan u pokrivanju klase odnosno koliko primera za testiranje iz date klase klasifikator može da prepozna:

$$\text{Odziv} = \frac{SP}{SP + LN}$$

Tačnost koja je korisna u slučajevima kada su klase iste ili slične veličine:

$$\text{Tačnost} = \frac{SP + SN}{SP + SN + LP + LN}$$

5.2 Testiranje

Za testiranje Pocketsphinx biblioteke, uzeta je ključna reč na koju sistem treba da reaguje i 4 reči slične toj ključnoj reči na koje sistem ne treba da se aktivira. Testiranje je pokazalo da je:

$$SP = 44, SN = 60, LP = 0, LN = 16$$

Odatle je lako izračunati preciznost, odziv i tačnost:

$$\text{Preciznost} = 1, \quad \text{Odziv} = 0.74, \quad \text{Tačnost} = 0.87$$

Međutim, testiranje je izvršeno na desktop računaru. Kada je ceo projekat pokrenut na Raspberry Pi 2, ispostavilo se da Pocketsphinx biblioteka ima vrlo primetno kašnjenje, zbog čega je dalje testiranje bilo nemoguće. Zatim je odvojen samo modul za prepoznavanje

ključne reči, što je uslovilo da kašnjenje bude smanjeno, ali i dalje ne dovoljno da bi testiranje bilo moguće.

Sledeća tačka testiranja predstavlja testiranje odziva Google Speech servisa što je urađeno merenjem vremena od slanja datoteke do prijema odgovora servisa. Zatim je mereno vreme od prijema odgovora, do same objave sadržaja, odnosno fizičke realizacije komande. Rezultati su dati u narednoj tabeli:

T_od_G [s]	T_za_E [s]	I_p
1,896	1,175	+
1,488	1,164	+
1,792	1,175	+
1,453	1,170	+
1,513	1,177	+
1,559	1,191	+
1,523	1,181	+
2,027	1,174	+
1,658	-	-
1,795	1,241	+
2,105	1,183	+
1,133	-	-
1,651	1,189	+
1,815	1,171	+
2,622	1,184	+
1,577	1,188	+
1,487	1,166	+
1,486	1,168	+
1,946	-	-
1,644	-	-

Tabela 5.1 Rezultati testiranja kvaliteta

Prva kolona u Tabeli 5.1 predstavlja vreme koje je potrebno da Google Speech servis prepozna izgovorenu komandu. Druga kolona predstavlja vreme potrebno za obradu odgovora dobijenog od strane Google Speech servisa i objavu formiranog sadržaja. U trećoj koloni se nalazi indikator uspešnog prepoznavanja od strane Google Speech servisa.

Kao što se vidi iz Tabele 5.1 vreme prepoznavanja komande ne unosi preveliko kašnjenje u sistem. Rezultati zavise i od dužine izdate komande, na primer, ako je komanda dužine 15 sekundi izmereno vreme prepoznavanja Google Speech servisa je 5,137 sekundi.

Komanda se izvršava približno jednu sekundu posle primljenog odgovora od strane servisa i uvek se uspešno izvršava ako je servis dobro uradio prepoznavanje izdate komande od strane korisnika. Google Speech servis ima uspešnost prepoznavanja od 80% što je zadovoljavajuće za praktične svrhe.

6. Zaključak

Cilj projekta je bio napraviti uređaj koji bi zamenio centralni uređaj OBLO sistema. Ovako zaseban uređaj, za razliku od postojećih komercijalnih rešenja, nudi mogućnost dodavanja novih funkcionalnosti, optimizacije postojećih i uticanje na svaki od modula i delova projekta pojedinačno. Problem se javlja u tome što je centralni uređaj namenski računar vrlo ograničenih kapaciteta i resursa, jer određeni delovi projekta zahtevaju jači procesor, ili namenski DSP procesor, kao i više radne memorije.

Projekat je testiran na opštenamenskom desktop računaru i na njemu nema većih kašnjenja, i bitnijih problema koje bi smanjile funkcionalnost. Međutim, testiranje na Raspberry Pi 2, pokazalo se problematično upravo zbog ograničenosti procesora i radne memorije. Problem bi rešila upotreba dva Raspberry PI 2 računara na kojima bi se odvojeno izvršavale dve celine projekta. Komunikacija, odnosno, razmena određenih zastavica izvršavala bi se ili serijskim putem ili preko Ethernet kabela kako bi se dovoljno smanjilo kašnjenje prouzrokovano odvajanjem projekta. Upotreborom DSP procesora jačih kapaciteta u kombinaciji sa opštenamenskim procesorom rešilo bi se kašnjenje prouzrokovano obradom Pocketsphinx biblioteke nad odbircima signala (govora). Samim tim, ostvarila bi se mogućnost upotrebe nizova mikrofona radi što bolje predstave signala, što bi rešilo probleme pogrešnih pogodaka od strane Pocketsphinx biblioteke, odnosno, poboljšalo bi i prepoznavanje Google Speech servisa.

7. Literatura

- [1] ZIGBEE, Alliance. Zigbee specification. ZigBee document 053474r13, 2006
- [2] C. Paetz, “Z-Wave Basics: Remote Control in Smart Homes”, 2013
- [3] M. Tucić, “Protokol za razmenu poruka u sistemima pametnih kuća”, Univerzitet u Novom Sadu, Fakultet Tehničkih Nauka, 2016
- [4] POCO biblioteka, [Online] Available: <https://pocoprotect.org/docs/>
- [5] Phil Blunsom, “Hidden Markov Models”, Avgust 2004
- [6] M. Kresoja, “Generacija scenarija pomoću skrivenih modela Markova”, Univerzitet u Novom Sadu, Prirodno-matematički Fakultet, 2011
- [7] G. D. Forney, “The Viterbi algorithm”, Proceedings of the IEEE, vol. 61, pp. 268–277, 1973.
- [8] C. Bishop “Pattern Recognition and Machine Learning”, pp. 629, 2006
- [9] D. Huggins-Daines, M. Kumar, et al., “Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices”, *Proc. of IEEE ICASSP*, 2006
- [10] F. Chabaud, A. Joux, “Differential collisions in SHA-o”, *CRYPTO*, pp56-71, 1998
- [11] M. Dworkin, “Recommendation for Block Cipher Modes of Operation”, *NIST*, april, 2017
- [12] [RFC2898] B. Kaliski, “PKCS #5: Password-Based Cryptography Specification Version 2.0”, septembar 2000
- [13] [RFC4648] S. Josefsson, “The Base16, Base32, and Base64 Data Encodings”, oktobar 2006
- [14] J. Graovac, “Prilog metodama klasifikacije teksta: Matematički modeli i primene”, Univerzitet u Beogradu, Matematički fakultet