



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД
Департман за рачунарство и аутоматику
Одсек за рачунарску технику и рачунарске комуникације**

ЗАВРШНИ (BACHELOR) РАД

Кандидат: Александар Лазић
Број индекса: РА 104/2014

Тема рада: Интеграција подршке за *Google Assistant* у апликацији за гледање телевизије на оперативном систему Андроид

Ментор рада: Доц. др Милан Бјелица

Нови Сад, јун 2018



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 Лазивић
Ментор, МН:	Доц. др Милан Бјелица
Наслов рада, НР:	Интеграција подршке за <i>Google Assistant</i> у апликацији за гледање телевизије на оперативном систему Андроид
Језик публикације, ЈП:	Српски / латиница
Језик извода, ЈИ:	Српски
Земља публикавања, ЗП:	Република Србија
Уже географско подручје, УГП:	Војводина
Година, ГО:	2018
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)	7/52/33/2/35/0/0
Научна област, НО:	Електротехника и рачунарство
Научна дисциплина, НД:	Рачунарска техника
Предметна одредница/Кључне речи, ПО:	Андроид, Гугл асистент, дигитална телевизија, гласовна команда, интеграција
УДК	
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, ВН:	
Извод, ИЗ:	Велика већина алата за препознавање гласовних команди генеришу резултат наредбе у виду слободног текста или структурираног објекта. Како би се тај резултат искористио у постојећим ТВ апликацијама, неопходно је детектовати шаблоне који одговарају наредбама попут промене канала, мењања јачине звука и сл. У овом раду је представљено једно решење коришћења ТВ апликације, уз помоћ Гугл асистента као алата за гласовно препознавање, за оперативни систем Андроид.
Датум прихватања теме, ДП:	
Датум одбране, ДО:	
Чланови комисије, КО:	Председник: Проф. др Иштван Пап
	Члан: Доц. др Јелена Ковачевић
	Члан, ментор: Доц. др Милан Бјелица
	Потпис ментора



UNIVERSITY OF NOVI SAD • FACULTY OF TECHNICAL SCIENCES
21000 NOVI SAD, Trg Dositeja Obradovića 6

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 Lazić
Mentor, MN :	Doc. dr Milan Bjelica
Title, TI :	Google Assistant integration in Live Channels application for Android OS
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 :	2018
Publisher, PB :	Author's reprint
Publication place, PP :	Novi Sad, Dositeja Obradovica sq. 6
Physical description, PD : <small>(chapters/pages/ref./tables/pictures/graphs/appendixes)</small>	7/52/33/2/35/0/0
Scientific field, SF :	Electrical Engineering
Scientific discipline, SD :	Computer Engineering, Engineering of Computer Based Systems
Subject/Key words, S/KW :	Android, Digital Television, Google Assistant, voice command, integration
UC	
Holding data, HD :	The Library of Faculty of Technical Sciences, Novi Sad, Serbia
Note, N :	
Abstract, AB :	<p>The majority of existing speech recognition tools provide the result of the speech processing as a free form textual output or structured form textual output. In order to use obtained outputs in an existing TV applications, it is necessary to detect patterns that correspond to commands such as channel up/down, volume up/down, mute/unmute, etc.</p> <p>This paper presents one solution for using TV application with the Google Assistant support for Android OS.</p>
Accepted by the Scientific Board on, ASB :	
Defended on, DE :	
Defended Board, DB :	President: Prof. dr Ištvan Pap
	Member: Doc. dr Jelena Kovačević
	Member, Mentor: Doc. dr Milan Bjelica
	Mentor's sign

Zahvalnost

Prvenstveno se zahvaljujem porodici i prijateljima na pruženoj podršci tokom celokupnog školovanja. Takođe se zahvaljujem institutu RT-RK na pruženoj mogućnosti za izradu ovog rada. Hvala svim profesorima i asistentima na stečenom znanju. Posebno se zahvaljujem tehničkom mentoru Dejanu Nađu na savetima i pomoći tokom izrade završnog rada, kao i mentoru doc. dr Milanu Bjelici.



УНИВЕРЗИТЕТ У НОВОМ САДУ ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



SADRŽAJ

1. Uvod.....	1
2. Teorijske osnove.....	3
2.1 Digitalna televizija.....	3
2.2 Android operativni sistem.....	4
2.2.1 Arhitektura Androida.....	5
2.2.1.1 Jezgro kernela.....	6
2.2.1.2 Sistemske (nativne) aplikacije.....	6
2.2.1.3 Izvršno radno okruženje.....	6
2.2.1.4 Okruženje za razvoj aplikacija.....	6
2.2.1.5 Aplikativni sloj.....	6
2.2.2 Android Open Source Project (AOSP).....	7
2.3 Android programska podrška za TV uređaje – TIF.....	7
2.3.1 Komponente programske podrške za TV uređaje.....	8
2.3.1.1 TV aplikacija.....	8
2.3.1.2 TV snabdevač.....	9
2.3.1.3 TV ulazni rukovalac.....	10
2.3.1.4 TV ulaz.....	11
2.4 Google Asistent (GA).....	12
2.4.1 Akcije na Guglu (Actions on Google).....	13
2.4.2 Princip rada <i>Google</i> Asistenta, aplikacije i akcija.....	14
2.5 Aplikacija za emitovanje kanala uživo (<i>Live Channels App</i>).....	16
3. Koncept rešenja.....	17
3.1 Arhitektura rešenja.....	17

3.1.1	Obučavanje <i>Google</i> Asistenta	18
3.1.2	Integracija <i>Google</i> Asistenta u TV aplikaciju	19
3.1.3	Izvršavanje glasovnih naredbi korisnika unutar TV aplikacije	19
4.	Programsko rešenje.....	20
4.1	Spisak modula.....	20
4.1.1	Modul za obučavanje GA	21
4.1.2	Modul za integraciju GA u TV aplikaciju	23
4.1.3	Modul za izvršavanje glasovnih komandi	26
4.1.3.1	void onResult(AIResponse result)	26
4.1.3.2	void onError(AIError error)	29
4.1.3.3	void onListeningFinished()	30
4.1.3.4	void OnListeningCanceled()	30
5.	Rezultati.....	31
5.1	Ispitivanje uspešnosti svakog od slučaja korišćenja	31
5.2	Ispitivanje vremena odziva <i>Google</i> Asistenta.....	33
5.3	Ispitivanje uspešnosti prepoznavanja komandi od strane asistenta	34
6.	Zaključak.....	36
7.	Literatura.....	37

SPISAK SLIKA

<i>Slika 2.1</i> - Ilustracija digitalne televizije.....	3
<i>Slika 2.2</i> - Mapa sveta po standardima.....	4
<i>Slika 2.3</i> - Arhitektura operativnog sistema Android.....	5
<i>Slika 2.4</i> - Android programska podrška za TV uređaje - TIF.....	7
<i>Slika 2.5</i> - Dijagram TV snabdevača.....	10
<i>Slika 2.6</i> - Dijagram TV ulaza treće strane.....	12
<i>Slika 2.7</i> - Primer konverzacije između korisnika i asistenta.....	13
<i>Slika 2.8</i> - Primer izgleda JSON zahteva.....	15
<i>Slika 2.9</i> - <i>Live Channels</i> aplikacija, promena kanala.....	16
<i>Slika 3.1</i> - Dijagram toka izvršavanja programa.....	18
<i>Slika 3.2</i> - Dijagram funkcionisanja TV aplikacije u kojoj je integrisan <i>Google Asistent</i>	20
<i>Slika 4.1</i> - Namere za kreiranje akcija.....	21
<i>Slika 4.2</i> - Očekivan unos korisnika prilikom komande „prebaci na željeni kanal“.....	22
<i>Slika 4.3</i> - Očekivan unos korisnika prilikom ostalih ključnih komandi.....	22
<i>Slika 4.4</i> - Testiranje rada GA.....	23
<i>Slika 4.5</i> - Deo <i>AndroidManifest.xml</i> datoteke.....	24
<i>Slika 4.6</i> - Neophodne dinamičke biblioteke.....	24
<i>Slika 4.7</i> - Konfiguracija asistenta sa aplikacijom.....	25
<i>Slika 4.8</i> - <i>OnKeyDown</i> metoda.....	26
<i>Slika 4.9</i> - Parsiranje parametara.....	27

<i>Slika 4.10</i> - Slučaj korišćenja <i>Channel up</i>	28
<i>Slika 4.11</i> - Slučaj korišćenja pojačavanja tona.....	28
<i>Slika 4.12</i> - Slučaj korišćenja potpunog utišavanja tona.....	28
<i>Slika 4.13</i> - Slučaj korišćenja <i>tuneToChannel</i>	29
<i>Slika 4.14</i> - Izgled ekrana sa porukom o grešci.....	30
<i>Slika 5.1</i> - Izgled aplikacije prilikom promene kanala.....	32
<i>Slika 5.2</i> - Grafički prikaz na ekranu prilikom pojačavanja tona.....	33
<i>Slika 5.3</i> - Grafički prikaz na ekranu prilikom potpunog utišavanja tona.....	33
<i>Slika 5.4</i> - Procenat uspešnosti prepoznavanja komandi.....	34

SPISAK TABELA

<i>Tabela 1</i> - Spisak akcija za obučavanje GA.....	18
<i>Tabela 2</i> - Moduli programskog rešenja.....	20
<i>Tabela 3</i> - Rezultati uspešnosti izvršavanja naredbi	32
<i>Tabela 4</i> - Rezultati merenja vremena odziva za komande.....	34

SKRAĆENICE

AI	- <i>Artificial Intelligence</i> , Veštačka inteligencija
API	- <i>Application Programming Interface</i> , Aplikativna programska sprega
GA	- <i>Google Assistant</i> , Gugl asistent
HTTPS	- <i>Hypertext Transfer Protocol Secure</i> , Komunikacioni protokol za sigurnu komunikaciju.
JSON	- <i>JavaScript Object Notation</i> , Standard za tekstualni opis podataka
STB	- <i>Set-Top Box</i> , Uređaj za prijem TV signala
TIF	- <i>TV Input Framework</i> , Android programska podrška TV prijemnika
TV	- <i>Television</i> , televizija
URI	- <i>Uniform Resource Identifier</i> , Jedinostveni identifikator resursa
URL	- <i>Uniform Resource Locator</i> , Jedinostveni lokator resursa

1. Uvod

U ovom radu je opisan problem, koncept rešenja i realizacija podrške *Google Asistentu (GA)* u *Google Live Channels* aplikaciji za osnovne funkcionalnosti promene kanala, jačine zvuka, potpuno utišavanje i vraćanje na jačinu tona koja je prethodila potpunom utišavanju. Neophodno je da korisnik razgovetno izgovori kompletnu komandu na engleskom jeziku. Podrška je realizovana za proširenje postojećeg prilagodnog sloja za Synaptics Sequoia platformu na najnovijem čipu Marvell BG5CT u okviru operativnog sistema Android. Prikazana je spregra između Java programskog jezika i GA, virtuelnog asistenta u oblaku (eng. *Cloud Computing*).

Za uspešnu izvedbu ovog rada neophodno je poznavati koncepte i principe rada računarstva u oblaku [1] i digitalne televizije [2].

Računarstvo u oblaku predstavlja isporuku resursa kao usluge krajnjim korisnicima, najčešće posredstvom Interneta, odnosno HTTPS protokola. Tako korisnici putem „pametnih“ uređaja pristupaju aplikacijama, gde se programska podrška i korisnički podaci nalaze na serverima na udaljenoj lokaciji.

Televizija je telekomunikacioni medijum za slanje i prijem pokretnih slika i zvuka. Trenutno predstavlja dominantan izvor zabave u domaćinstvima prema brojnim istraživanjima [3] sprovedenim u svetu sa udelom od skoro 80%. U stopu je prati Internet, dok su radio i novine najmanje zastupljeni. Upravo taj spoj televizije i interneta predstavlja dobitnu kombinaciju za korisnike čiji zahtevi stalo rastu kao i potreba da im digitalni televizijski prijemnik u što većoj meri zameni „pametni“ telefon.

Moderna tehnologija teži ga povezivanju i integraciji digitalnih televizijskih prijemnika sa uređajima poput „pametnih” telefona, računara, tableta, ali isto tako i sa servisima u oblaku poput servisa *Amazon Alexa*, *Siri* ili *Google Assistant*. Ovi asistenti osim što pružaju ugrađene funkcionalnosti, dozvoljavaju i razvoj sopstvenih funkcionalnosti. Sam napredak tehnologije omogućio je i lakšu integraciju sa njima, što u mnogome poboljšava korisničko iskustvo i olakšava svakodnevni život i osnovne ljudske potrebe.

Upravo je u okviru ovog rada GA uspešno integrisan u već postojeću Guglovu *Live Channels* aplikaciju, što će biti opisano u nastavku. Pomoću njega je korisniku olakšana interakcija sa digitalnim televizijskim prijemnikom, gde može da upravlja njime koristeći glasovne komande za osnovne funkcionalnosti.

Rad se sastoji iz sedam poglavlja. U prvom se nalazi kratak opis rada i osnovni podaci o njemu, odnosno kraći uvod.

U drugom poglavlju ovog rada je data kratka teorijska osnova o samom operativnom sistemu Android i načinu na koji on funkcioniše. Takođe, date su osnove o digitalnoj televiziji kao i o Android programskoj podršci za TV uređaje (eng. *TV Input Framework – TIF*). Ukratko je opisan i sam *Google Asistent* kao i akcije na Guglu (eng. *Actions on Google*).

U trećem poglavlju predstavljeno je rešenje gde je izneseno idejno rešenje rada podeljeno na samu aplikaciju, *Google Asistent* zasebno i njihovu integraciju na kraju.

Četvrto poglavlje predstavlja programsko rešenje, odnosno realizaciju funkcija i metoda potrebnih za integraciju sa asistentom i izvršavanje korisničkih željenih komandi.

U petom poglavlju su predstavljeni rezultati testiranja i verifikacije. Opisani su dobijeni rezultati vremena odziva aplikacije kao i preciznosti prepoznavanja govornih komandi od strane GA.

Pretposlednje poglavlje sadrži kratak zaključak o onome što je urađeno u radu, dok je na samom kraju dat spisak korišćene literature tokom izrade ovog rada.

2. Teorijske osnove

U ovom poglavlju će biti date teorijske osnove i detaljniji opis pojmova neophodnih za razumevanje tematike koja je obrađena ovim radom.

2.1 Digitalna televizija

Sam pojam televizije predstavlja prenos slike na daljinu. Kako je tehnologija napredovala i razvijala se tako se i televizija razvijala te je vremenom analogna televizija zamenjena digitalnom. Digitalna televizija predstavlja prenos audio i video zapisa, kao i drugih informacija poput TV vodiča, informacija o vremenu i datumu, programima i sl, u digitalnom formatu.

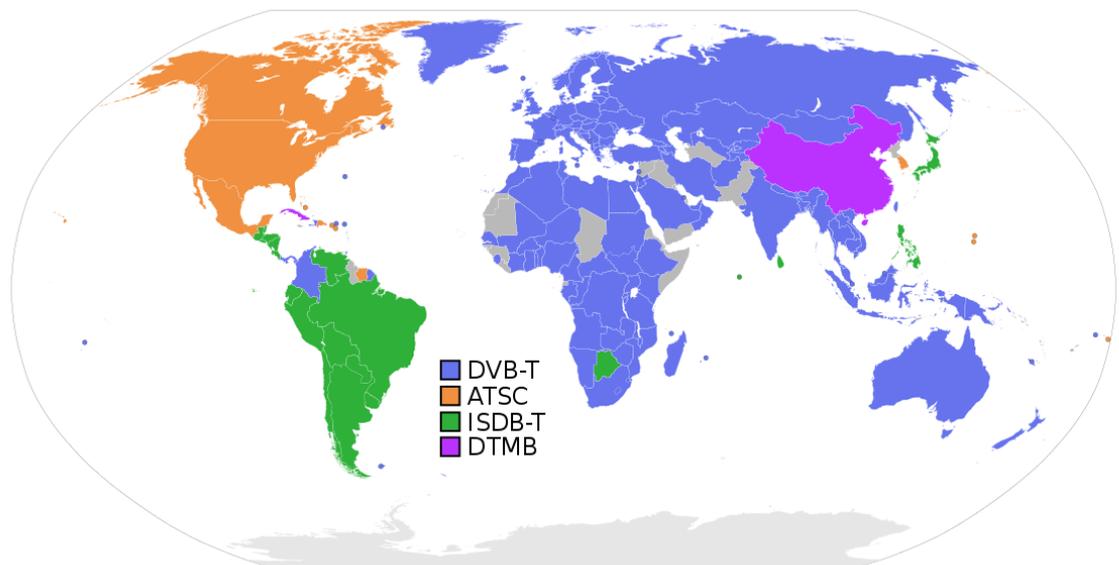


Slika 2.1 – Ilustracija digitalne televizije

Ono po čemu se analogna i digitalna televizija najviše razlikuju je kvalitet slike i zvuka, u korist digitalne. Druga velika prednost je veći propusni opseg kod digitalne televizije čime

se dovodi do uštede prilikom prenosa TV servisa i drugih informacija, koristeći jednu frekvenciju. Treća, možda i najveća prednost iz ugla korisnika, je moguća interakcija sa istim. Mnoštvo Android aplikacija pisane u Java programskom jeziku, WEB aplikacije i pristup internetu omogućavaju potpunu interakciju sa korisnikom, koji može da prilagodi sadržaj svojim potrebama i preferencijama. Neki od najznačajnijih standarda u digitalnoj televiziji su:

- DVB – najrasprostranjeniji, koristi se u Evropi i većem delu sveta;
- OCAP – koristi se u SAD;
- ATSC – standardi koji se primenjuju u Severnoj Americi, Srednjoj Americi (Meksiko) i Južnoj Koreji;
- DTMB – na dalekom istoku u Kini se koristi ovaj standard;
- ISDB – standard koji je vezan na Japan.



Slika 2.2 – Mapa sveta po odabranom DTV standardu

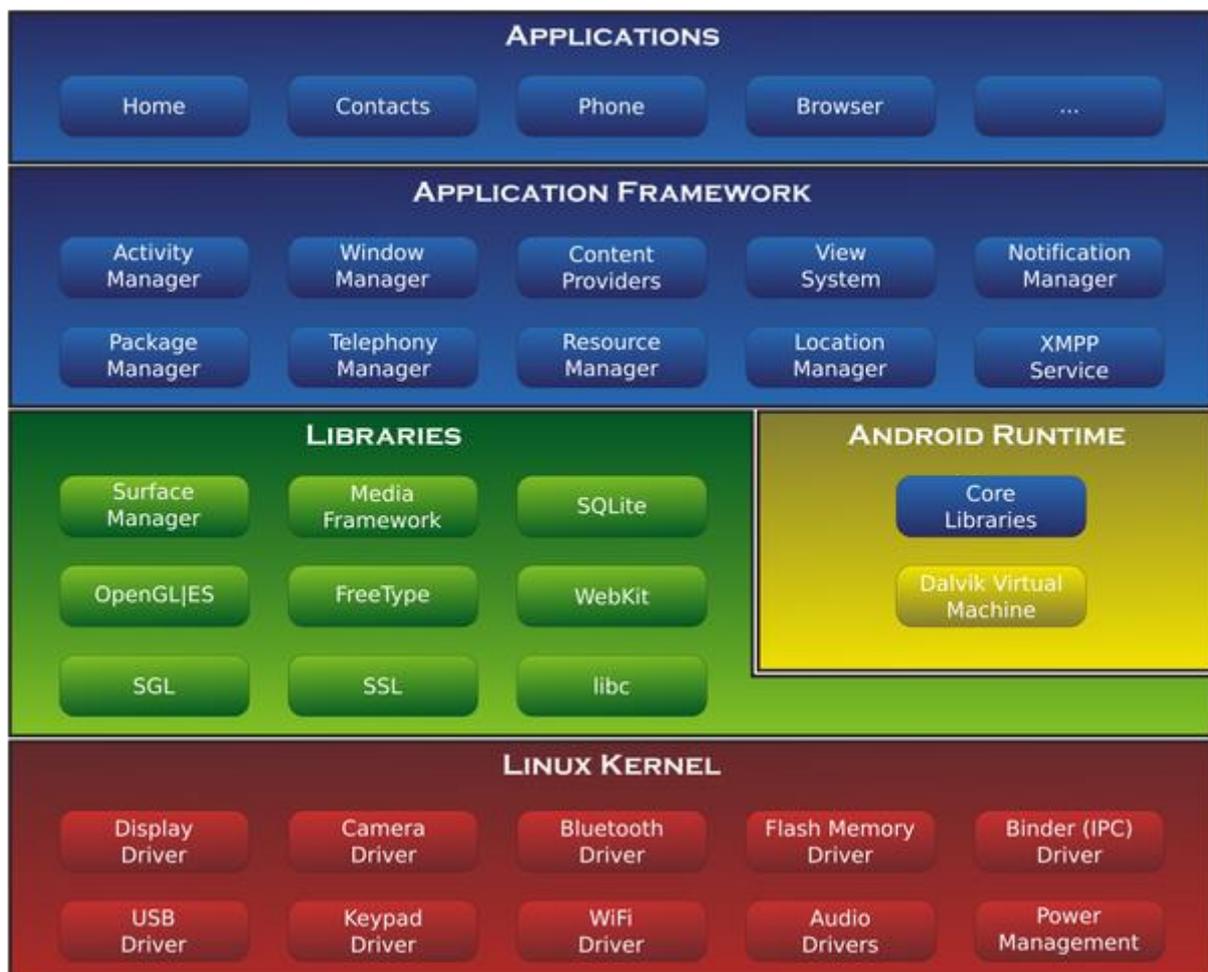
2.2 Android operativni sistem

Android [4] je jedan od najrasprostranjenijih operativnih sistema u svetu, namenjen za namenske uređaje, pre svega mobilne telefone i tablete. U poslednjih nekoliko godina dolazi do velike ekspanzije ove platforme, posebno u primeni kod „pametnih“ satova, STB uređaja, ali i televizora. Ovo se, pre svega, može prepisati činjenici da je ovaj operativni sistem otvorenog koda (eng. *Open source*) dostupan pod *Apache* licencom, koja dozvoljava izmenu i distribuciju programske podrške od strane proizvođača uređaja. Iz tog razloga su Android aplikacije nezavisne od fizičke arhitekture, tj. mogu se izvršavati na različitim platformama.

2.2.1 Arhitektura Androida

Android stek se može podeliti u pet slojeva programske podrške, što se može videti i sa slike 2.3. Slojevi su (od nižeg ka višem) :

- Linuks jezgro;
- Sistemske (nativne) biblioteke;
- Izvršno radno okruženje;
- Okruženje za razvoj aplikacija;
- Aplikativni sloj.



Slika 2.3 - Arhitektura operativnog sistema Android

2.2.1.1 Jezgro kernela

Ovo je najniži sloj Android steka i na tom jezgru je i zasnovan sam Android. Jezgro kernela predstavlja most između programske podrške i fizičke arhitekture, odnosno predstavlja njenu apstrakciju. Ovaj sloj je toliko važan iz razloga što se u njemu nalaze moduli za upravljanje fizičkom arhitekturom, memorijom, procesima, mrežnim komunikacijama i dr.

2.2.1.2 Sistemske (nativne) aplikacije

Iznad sloja jezgra kernela nalazi se sloj sistemskih biblioteka. One su implementirane u programskom jeziku C i C++. Pored nativnih biblioteka, Android obuhvata i skup osnovnih biblioteka koje sadrže veći deo funkcionalnosti koje sadrže biblioteke u Java programskom jeziku. Neke od važnijih biblioteka ovog sloja su :

- Surface Manager – biblioteka zadužena za iscrtavanje različitih grafičkih elemenata;
- SQLite – biblioteka koja pruža podršku za upravljanje bazama podataka;
- WebKit – pruža podršku za pregledanje sadržaja na Internetu.

2.2.1.3 Izvršno radno okruženje

Izvršno radno okruženje se nalazi u istom sloju kao i sistemske biblioteke i od ključne je važnosti za razvoj aplikacija višeg nivoa u Java programskom jeziku. Te aplikacije se pokreću posredstvom virtuelne mašine koja je specijalno dizajnirana za Android operativni sistem i optimizovana za namenske uređaje, čiji resursi su poprilično ograničeni. Najpoznatije virtuelne mašine za Android su *ART* i *Dalvik*.

2.2.1.4 Okruženje za razvoj aplikacija

Ovo je sloj pisan u Java programskom jeziku i neophodan je jer sadrži skup biblioteka i sistemskih aplikacija za pokretanje i korišćenje aplikacija.

2.2.1.5 Aplikativni sloj

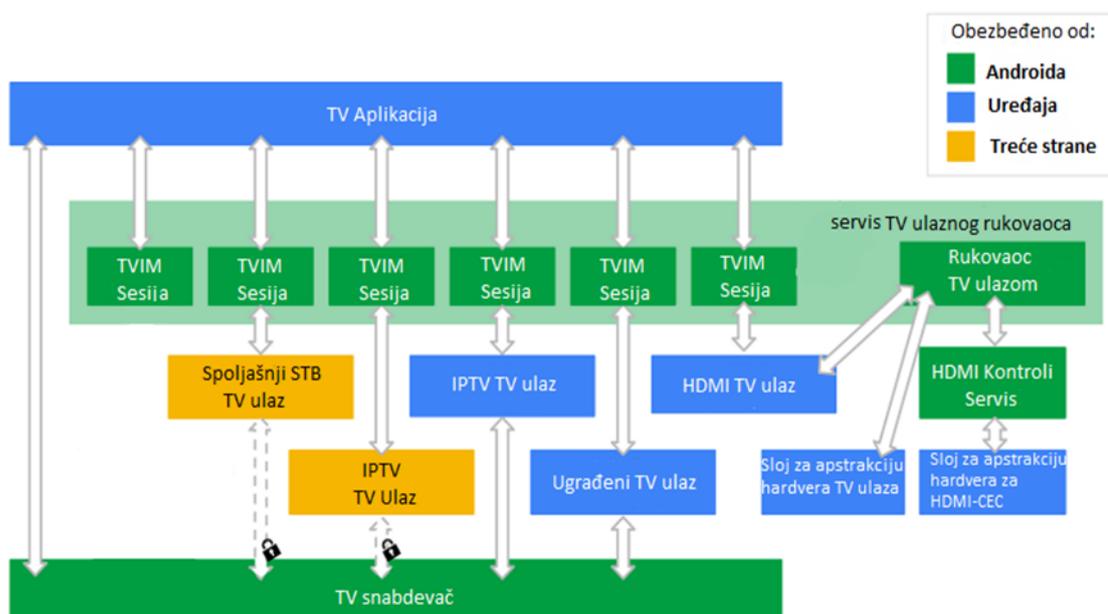
Nalazi se na vrhu steka i sastoji se od korisničkih aplikacija. To je jedini deo steka vidljiv korisniku i pomoću njega se apstrahuju svi niži slojevi.

2.2.2 Android Open Source Project (AOSP)

Android ima aktivnu zajednicu programera koji koriste projekat otvorenog koda (eng. Android Open Source Project - AOSP) da bi razvili i proizveli lične verzije ovog operativnog sistema. Izdanja koja napravi ova zajednica obično donose nove dodatke i nadogradnje uređajima brže nego što bi to uradio zvanični proizvođač, sa uporedivim nivoom kvaliteta. Takođe, oni nastavljaju sa pružanjem podrške za zastarele uređaje koji više ne dobijaju zvanične nadogradnje ili spuštaju Android na uređaje koji su prvobitno bili namenjeni za neke druge operativne sisteme. Ova izdanja često pružaju korisniku sva prava i dozvole i sadrže izmene koje nisu obezbeđene od strane prodavca, kao što je mogućnost promene takta i napona procesora uređaja.

2.3 Android programska podrška za TV uređaje – TIF

Android programska podrška za TV uređaje (eng. *TV input framework, TIF* [5]) definiše aplikativnu programsku spregu i standardizuje način dopremanja emitovanog sadržaja do TV aplikacije. Ova sprega omogućava implementaciju TV ulaza preko kog je omogućena pretraga televizije uživo, kao i reprodukcija sadržaja iz DVB transportnog toka i/ili sa mrežnog posluživača. Programska podrška ne teži ka tome da implementira TV standarde ili regionalne zahteve, ali proizvođačima uređaja omogućava da lakše ispune regionalne digitalne TV zahteve, bez ponovne implementacije.



Slika 2.4 - Android programska podrška za TV uređaje – TIF

Na osnovu datog dijagrama i komponenti sa slike može se zaključiti da:

- TV aplikacija prikazuje sadržaj sa TV ulaza;
- Korisnik može da vidi aplikaciju i da bude u direktnom kontaktu sa njom;
- Aplikacija ne može direktno da komunicira sa TV ulazima već posredstvom rukovaoca TV sadržajem, koji identifikuje stanje TV ulaza za TV aplikaciju.

2.3.1 Komponente programske podrške za TV uređaje

Pomoću TIF-a, TV aplikacija pristupa ugrađenim modulima koji su isporučeni od strane proizvođača ili neke druge strane putem ulaznog TV rukovaoca. Kao što se može primetiti sa dijagrama slike 2.4, TIF se sastoji iz nekoliko celina:

- TV aplikacija (eng. *TV Application*) – služi za interakciju sa korisnikom, korisnik preko nje upravlja programskom podrškom za TV uređaje;
- TV snabdevač (eng. *TV Provider*) – predstavlja bazu podataka sa kanalima, programima i pratećim dozvolama;
- TV ulazni rukovalac (eng. *TV input manager*) – omogućava komunikaciju između TV ulaza i same TV aplikacije;
- TV ulaz (eng. *TV input*) – izvor televizijskog sadržaja;
- Ulazna TV fizička arhitektura (eng. *TV Input Hardware Abstraction Layer*) – omogućava pristup fizičkoj arhitekturi specifičnom za televiziju;
- Roditeljska kontrola (eng. *Parental Control*) – tehnologija koja omogućava blokiranje kanala;
- HDMI-CEC – tehnologija koja omogućava daljinsku kontrolu raznih uređaja putem HDMI-CEC poruka.

Svaka od ovih komponenti će biti detaljnije opisana u daljem tekstu.

2.3.1.1 TV aplikacija

Sistemska TV aplikacija prikazuje korisniku TV sadržaj. Referentna TV aplikacija (*Live Channels*) je obezbeđena zajedno sa Android platformom, koja se može menjati, proširivati ili koristiti takva kakva jeste.

Minimalno što sistemska aplikacija mora da podrži na verziji Android Nougat je:

1. Podešavanje i konfiguracije:

- Automatska detekcija TV ulaza;
- Menjanje kanala;
- Kontrola roditeljskih podešavanja;
- Dozvola TV ulazima da inicijalizuju postavke kanala.

2. Gledanje:

- Pristup i navigacija svim TV kanalima;
- Pristup informacionim podacima TV programa;
- Prikaz elektronskog programskog vodiča;
- Podrška za više zvučnih i prevod traka;
- Prikaz PIN zahteva roditeljske kontrole;
- Podrška za gledanje sadržaja ispočetka;
- Podrška za slika-u-slici režim rada;
- Rukovanje snimanjem digitalnog sadržaja i podrška za *TV recording* API;
- Popunjavanje rezultata pretrage za TV kanale i programe.

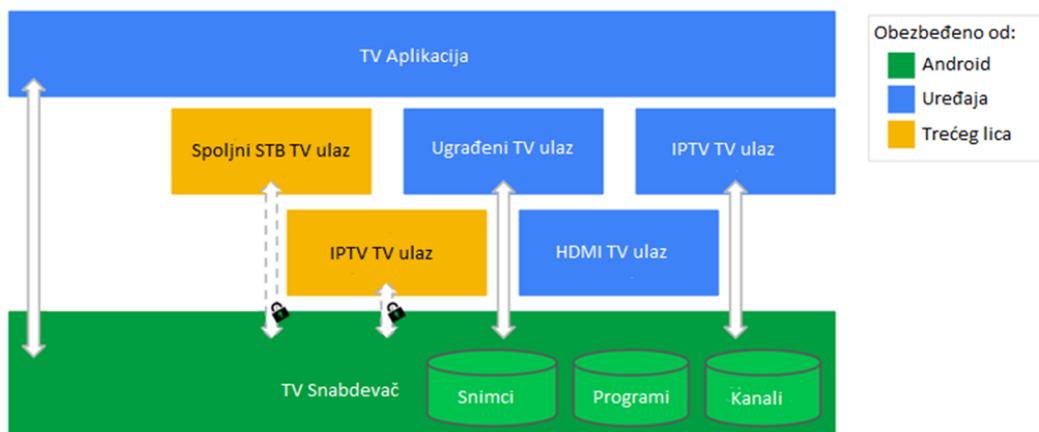
Ovaj skup funkcija će se proširivati sa Android verzijama, kako se i TIF API–ji budu proširivali. Da bi se proverilo da li aplikacija ima sve potrebne komponente postoji aplikacija CTS verifikator (eng. *Compatibility test suite*) koja obezbeđuje ispitivanje usklađenosti. Proizvođači moraju da implementiraju TV aplikaciju koja uključuje rezultate pretrage koji će biti uključeni u globalnu pretragu kako bi osigurali najbolje korisničko iskustvo.

Referentna Live TV aplikacija pruža implementaciju koja obezbeđuje rezultate samostalnih ulaza kao i rezultate ugrađenih ulaza.

2.3.1.2 TV snabdevač

TV snabdevač (eng. *TV Provider*) je baza podataka koja sadrži listu kanala, programa (EPG) i snimljenog sadržaja. On dobavlja podatke od strane TV ulaza. Da bi se obezbedila konzistentnost podataka dobavljenih od strane drugih TV ulaza, TV dobavljač rukuje odgovarajućim dozvolama kako bi TV ulazi mogli da vide samo svoje zapise, tj . da ne bi došlo do konflikta.

Mogućnost čitanja podataka iz baze podataka imaju samo aplikacije koje se nalaze u privilegovanoj sistemskoj particiji. Ostale samostalne aplikacije mogu da pristupaju samo poljima u bazi podataka koja su oni popunili. Kao dodatak standardnim poljima za kanale i programe, baza podataka TV snabdevača nudi i posebno polje za binarno velike objekte (eng. *BLOB-Binary large Object*) u svakoj tabeli koju TV ulazi koriste za skladištenje proizvoljnih podataka. U tim objektima se skladište prilagođene informacije poput frekvencija povezanog frekventnog odabirača, a mogu biti i obezbeđeni u nekoj drugoj formi. Takođe je dostupno i polje *column_searchable*, koje može da izuzme neke kanale iz pretrage, ali i da omogući glasovnu ili tekstualnu pretragu TV sadržaja, krucijalnu za ovaj rad. Sva ova polja su skrivena za korisnika, koji može da vidi samo ono što mu TV aplikacija, sistemske aplikacije ili TV ulazi prikažu.



Slika 2.5 – Dijagram TV snabdevača

2.3.1.3 TV ulazni rukovalac

TV ulazni rukovalac (eng. *TV Input Manager*) predstavlja centralnu sistemsku aplikativnu spregu (eng. *API, Application Programming Interface*) za ceo TIF. Njegov zadatak je da nadgleda celokupnu interakciju između aplikacija i TV ulaza, kao i da obezbedi funkcionalnost roditeljske kontrole. Sesija TV ulaznog rukovaoca mora biti kreirana jedan na jedan sa TV ulazom. Ovaj rukovalac može da dozvoli pristup aplikacijama TV ulazu tako da aplikacije mogu da:

- Naprave sesiju i rukuju slušaocima (eng. *Listeners*);
- Listaju TV ulaze i proveravaju njihove statuse.

Kada su sesije u pitanju, TV aplikacije mogu da se povežu na TV ulaze samo preko URI-ja koji su dodale u bazu podataka TV snabdevača, osim prolaznih TV ulaza na koje se mogu povezati koristeći *TvContract.buildChannelUriForPassthroughInput()*. TV ulazi koji

su obezbeđeni i imaju potpisan sertifikat od strane proizvođača uređaja ili druge aplikacije instalirane u sistemsku particiju imaju potpun pristup bazi podataka TV snabdevača. Ovaj pristup može da se iskoristi da se realizuju aplikacije koje će pretraživati sve dostupne TV kanale i programe.

Aplikacije mogu da naprave i prijave *TvInputCallback* koji će da pozove *android.media.tv.TvInputManager* ukoliko se promeni stanje TV ulaza ili se doda ili ukloni neki od TV ulaza. TV aplikacija može da reaguje kada se, na primer, neki TV ulaz isključi tako što će ga prikazati kao isključenog i zabraniti mogućnost biranja istog.

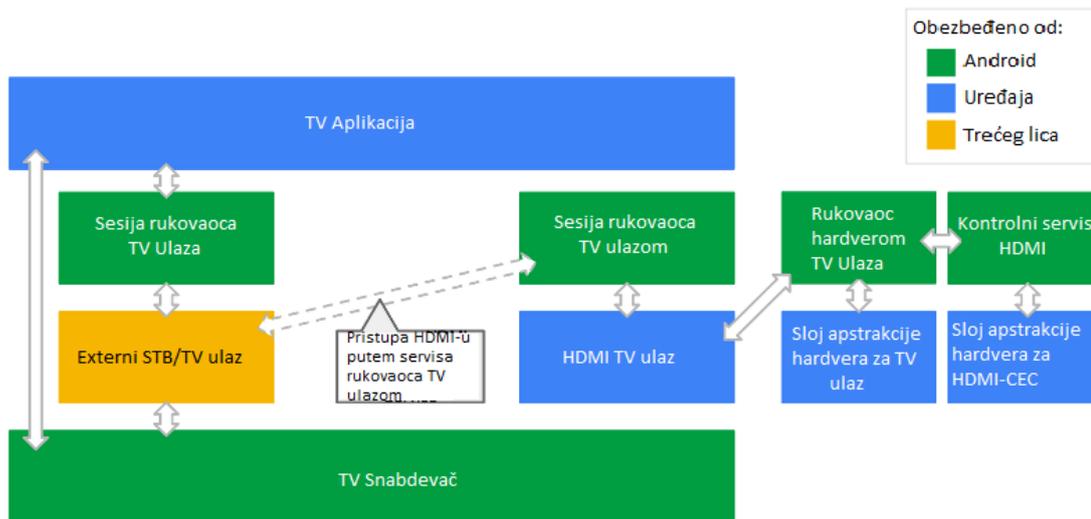
Rukovalac je takođe zadužen za apstrakciju TV aplikacije i TV ulaza. Standardna sprega omogućava da više proizvođača napravi svoje TV aplikacije, dok samostalnim TV ulazima pomažu da rade sa svim tim aplikacijama.

2.3.1.4 TV ulaz

TV ulazi (eng. *TV Inputs*) su zapravo Android aplikacije koje imaju *AndroidManifest.xml* fajl i instaliraju se. Android TV podržava:

- Unapred instalirane sistemske aplikacije
- Aplikacije sa potpisanim sertifikatom od strane proizvođača
- Samostalne TV ulaze (eng. *Third Party*).

Neki ulazi poput HMDI-ja (eng. *High-Definition Multimedia Interface*) ili ulaz frekventnog odabirača, mogu biti obezbeđeni samo od strane proizvođača jer direktno komuniciraju sa fizičkom arhitekturom. S druge strane, spoljni STB ili IPTV mogu biti u vidu APK (eng. *Android Package Kit*) datoteka. Kada se jednom preuzmu i instaliraju takve aplikacije, novi ulazi mogu biti izabrani u samoj TV aplikaciji.



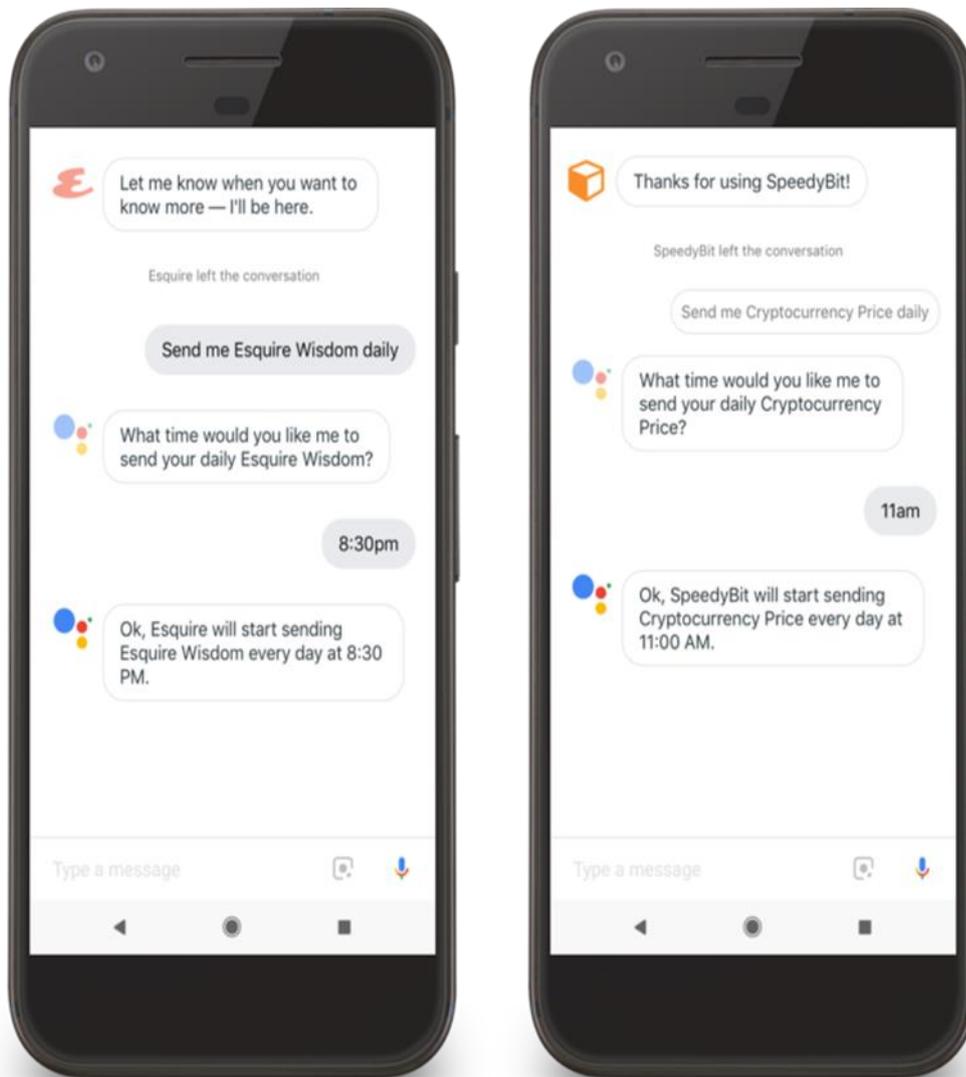
Slika 2.6 – Dijagram TV ulaza treće strane

Na slici 2.6 prikazan je dijagram gde se kao TV ulaz koristi spoljnji samostalni STB TV ulaz (eng. *Third party*). Najbitnija komponenta ovde je rukovalac ulaza preko kog TV ulaz pristupa jer ne može direktno da pristupi HDMI video izvoru koji pristiže. Pomoću rukovaoca spoljni STB TV komunicira sa HDMI TV ulazom i traži dozvolu da prikaže video putem HDMI-ja.

2.4 Google Asistent (GA)

U maju 2015. godine je predstavljen virtuelni, lični asistent koji poseduje veštačku inteligenciju AI (eng. *Artificial Intelligence*) – Google Asistent [6]. On pruža mogućnost dvosmerne komunikacije sa korisnikom, glasovnim ili tekstualnim putem. Korišćenjem teksta poboljšava se kvalitet konverzacije odnosno dobijenih odgovora, dok u slučaju glasovnih komandi, asistent koristi algoritme za procesiranje prirodnog jezika. Podržan je samo engleski jezik za sada. Kao i njegov prethodnik *Google Now*, može da izvršava mnoštvo raznoraznih komandi od podešavanja alarma, pronalaženja kontakta i slanje istom SMS poruke pa sve do pretraživanja Interneta. Dodata funkcionalnost u odnosu na prethodnika je da može da se nastavi dvosmerna komunikacija nakon zadavanja prvobitne komande, gde se pitanja i odgovori na istu temu nadovezuju. GA je integrisan u *Google home* uređaj, za potrebe „pametne“ kuće a vrlo brzo je pronašao primenu i u drugim namenskim uređajima. Takođe je integrisan i u Android uređaje sa operativnim sistemom Marshmallow (6.0 & 6.0.1) i Nougat(7.0) uključujući i Android Wear 2.0 [7].

Kako servis ne bi bespotrebno bio konstantno aktivan, uvedene su ključne reči za prepoznavanje: „*Ok Google*“ i „*Hey Google*“. Za potrebe rada, ove ključne reči su zamenjene pritiskom na dugme na daljinskom upravljaču, o čemu će više biti rečeno u narednim poglavljima.



Slika 2.7 – Primer konverzacije između korisnika i asistenta

2.4.1 Akcije na Guglu (Actions on Google)

Akcije na Guglu [8] predstavljaju dodatno proširenje za GA, odnosno, to je programerska platforma koja omogućava programerima da naprave svoje akcije ili prošire već postojeće. Drugim rečima, omogućeno je dodatno obučavanje GA za specifične radnje. Tako, na primer, vrlo lako neko ko i nema preveliko iskustvo u programiranju može da napravi sebi svojstvenu akciju i tako sebi olakša život.

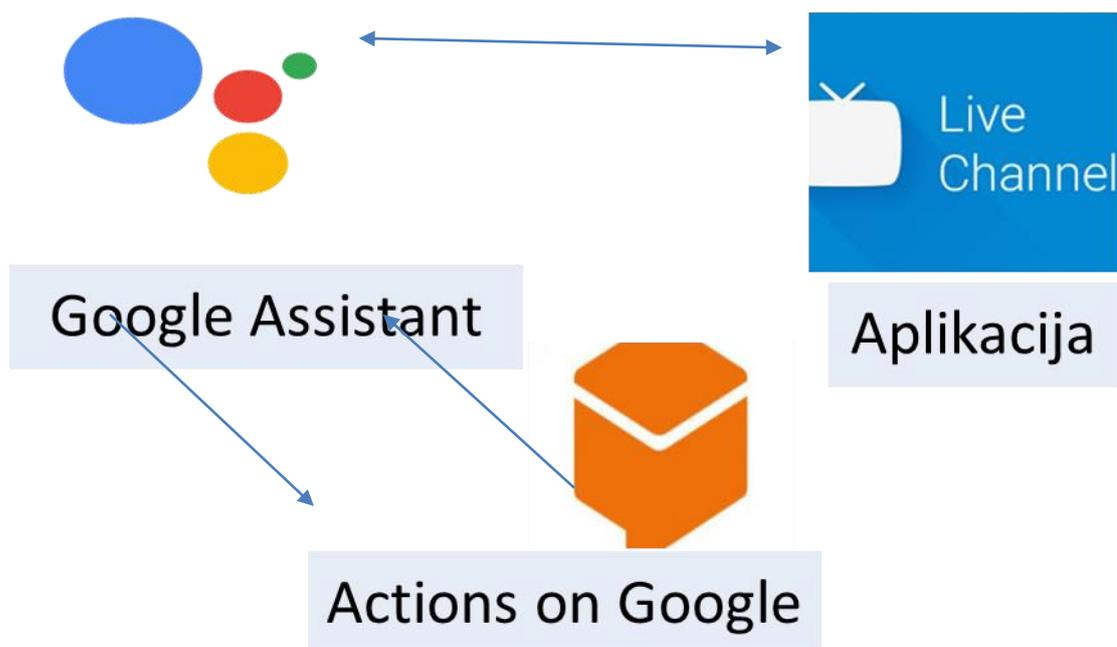
Još jedna velika pogodnost za celokupnu zajednicu programera kao i za sve ostale korisnike je da im akcija postaje dostupna za korišćenje ukoliko uspešno prođe verifikaciju i testiranje.

2.4.2 Princip rada *Google Asistenta*, aplikacije i akcija

Procesi koji se izvršavaju prilikom korisničkog zahteva za akcijom su :

- Korisnik šalje zahtev za nekom akcijom
- GA potražuje od platforme *Actions on Google* da pokrene željenu aplikaciju
- *Actions on Google* šalje zahtev na krajnju tačku izvršavanja i prosleđuje odgovor do GA
- GA prikazuje odgovor korisniku.
- Nakon toga, GA unos korisnika direktno prosleđuje aplikaciji i aplikacija direktno komunicira sa korisnikom, bez posredstva platforme.

Ovi procesi mogu se i grafički predstaviti sledećim dijagramom:



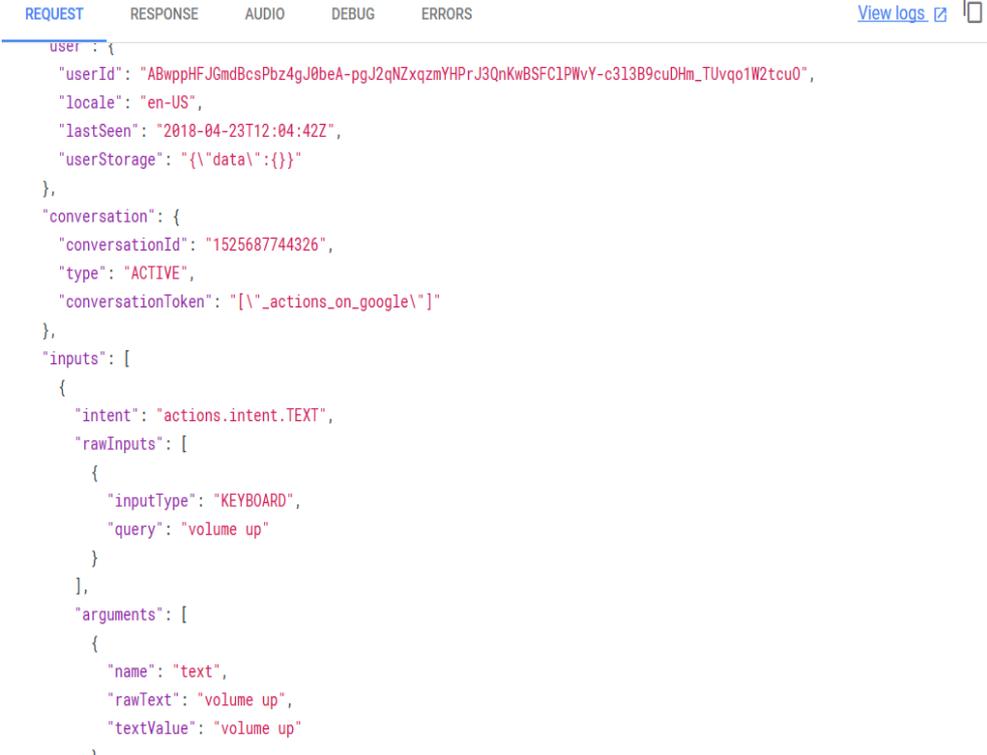
Postoje dva tipa sprega koji se koriste za razvoj akcija na ovoj platformi:

1. **API.AI** – najčešće korišćena sprega za razvijanje konverzijskih aplikacija. Vrlo lako rešava problem parsiranja i razumevanja ljudskog govora, što inače predstavlja izazov u programerskom svetu.

2. **Actions on SDK** – uglavnom se koristi za kraće konverzacije, gde se koristi ograničeni fond reči i mogućnost unosa od strane korisnika. Ovakvi tipovi ne zahtevaju robustno razumevanje jezika i obično pokrivaju jednostavne slučajeve korišćenja. Za razliku od API.AI, ovaj paket ne nudi pogodnosti poput generisanja i otpremanja akcijskih pokreta, te je neophodno samostalno to odraditi.

Akcija predstavlja ulaznu tačku i definiše model za pronalaženje i pokretanje aplikacije. Prilikom korisničkog zahteva, formira se JSON objekat koji u sebi sadrži određena polja sa informacijama o akciji. Dva neophodna polja su unosi (eng. *Inputs*) i konverzacije (eng. *Conversations*). Unutar unosa se definiše jedna ili više akcija, a unutar njih korisničke namere (eng. *Intents*) – funkcionalnost koje se obavljaju u zavisnosti od korisničkog unosa. Svaka od tih namera može imati svoju tačku izvršavanja.

Na slici 2.8 prikazan je deo JSON objekta prilikom korisničkog zahteva da se pojača ton na digitalnom televizijskom prijemniku.



```
REQUEST RESPONSE AUDIO DEBUG ERRORS View logs [icon]
user : {
  "userId": "ABwppHFJGmdBcsPbz4gJ0beA-pgJ2qNZxqzmYHPrJ3QnKwBSFC1PWvY-c313B9cuDHm_TUvqo1W2tcu0",
  "locale": "en-US",
  "lastSeen": "2018-04-23T12:04:42Z",
  "userStorage": "{\"data\":{}}"
},
"conversation": {
  "conversationId": "1525687744326",
  "type": "ACTIVE",
  "conversationToken": "[\\\"_actions_on_google\\\"]"
},
"inputs": [
  {
    "intent": "actions.intent.TEXT",
    "rawInputs": [
      {
        "inputType": "KEYBOARD",
        "query": "volume up"
      }
    ]
  }
],
"arguments": [
  {
    "name": "text",
    "rawText": "volume up",
    "textValue": "volume up"
  }
]
```

Slika 2.8 – Primer izgleda JSON zahteva

Korisničke namere (eng. *Intents*) predstavljaju jednostavne objekte koji bliže opisuju određeni proces. U akciji napravljenoj za ovaj rad, postoji osnovna namera *welcome_intent*, koja se poziva na početku akcije a kasnije se koriste namere *tune_to* ili *perform_action*, u zavisnosti od korisničkih želja. Najveći mogući broj namera u jednom akcijskom paketu je deset. Za dalji razvoj i integraciju sa aplikacijom, o čemu će više biti rečeno u narednom poglavlju, neophodno je još da *URL* koristi HTTPS protokol, kao i da se dobavi jedinstveni kod *URI* pomoću kog će aplikacija moći da prepozna akcijski paket.

2.5 Aplikacija za emitovanje kanala uživo (*Live Channels App*)

Google-ova aplikacija *Live Channels* služi za gledanje kanala uživo. Ona omogućava korisniku gledanje njegovih omiljenih filmova, sportskih programa, vesti, itd. Za potrebe izvršenja ovog zadatka potrebno je upoznati se sa API-jem ove aplikacije da bi se mogle izvršiti funkcionalnosti promene kanala ili jačine zvuka nakon što stigne određena komanda od strane *Google Asistenta*. Izgled aplikacije prikazan je na slici 2.9.



Slika 2.9 – *Live Channels* aplikacija, promena kanala

U ovoj aplikaciji je izvršena integracija *Google Asistenta*, o čemu će biti reči u daljem tekstu. Moguće ju je instalirati na uređaje sa Android Nougat i višim verzijama. Besplatno je dostupna na zvaničnoj Guglovoj prodavnici, sa preko milion preuzimanja do sada [9].

3. Koncept rešenja

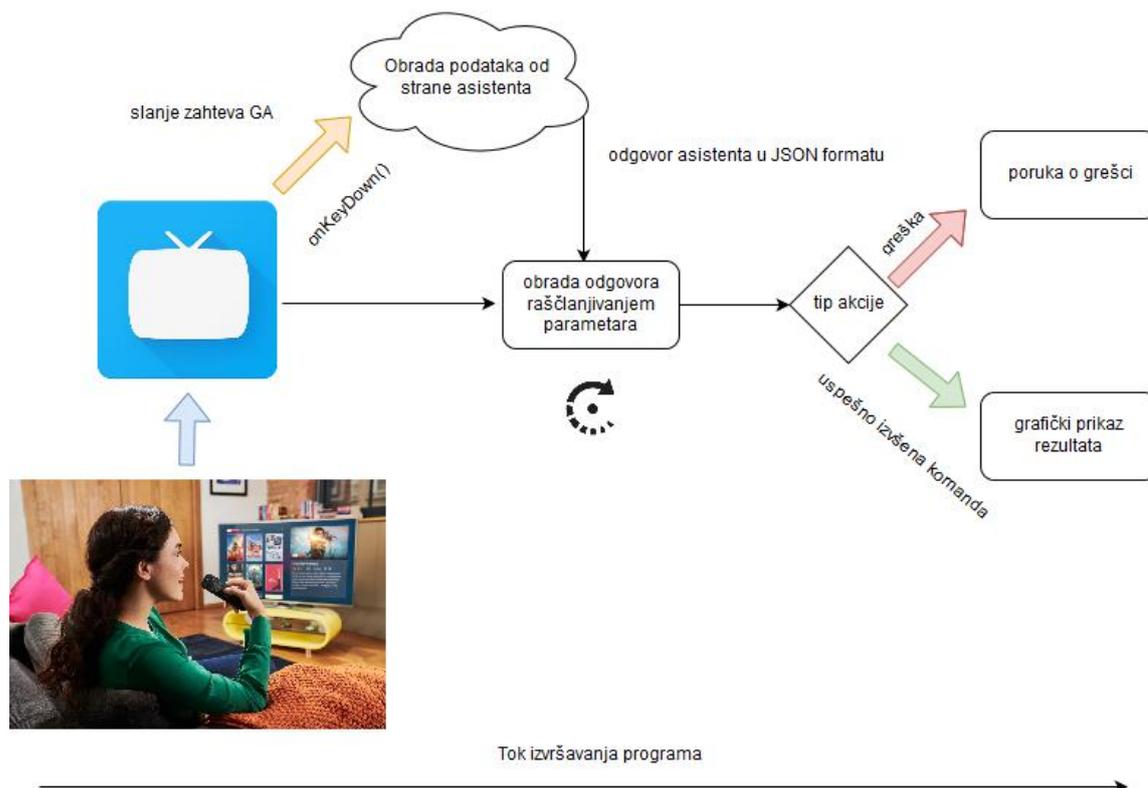
U ovom poglavlju biće izvršena analiza zadatog problema i opisano idejno rešenje. Integracija *Google Asistenta* u TV aplikaciju zahteva pre svega njegovo obučavanje za ulaze koje mu korisnik može zadati. S druge strane, potrebno je razumeti kako TV aplikacija radi, odnosno upoznati se sa njenim API-jem. Na samom kraju je potrebno napraviti vezu između ove dve celine.

3.1 Arhitektura rešenja

Na samom početku korisnik zadaje glasovnu komandu pritiskom na dugme na daljinskom upravljaču. Samim tim se aktivira i modul *Google Asistenta* koji počinje da sluša komandu. Nakon toga on komunicira sa posluživačem kako bi se izvršila određena akcija (*Actions on Google*). Kada asistent prepozna glasovnu komandu i akciju koja treba da se izvrši, on svoj odgovor šalje samoj TV aplikaciji. U TV aplikaciji je potrebno prihvatiti taj odgovor kada se završi period osluškivanja korisničke komande. Odgovor je najčešće u obliku JSON objekta i sadrži više parametara pa je potrebno isti i raščlaniti na više manjih ključnih reči u tekstualnom obliku (parsirati), kako bi se iz toga dobila jasna komanda koju treba izvršiti. Komandu je potom potrebno izvršiti ili prikazati poruku o grešci ako do nje dođe tokom obrade i izvršenja iste. Celokupno rešenje se može podeliti na sledeće tri celine:

- Obučavanje *Google Asistenta*;
- Integracija *Google Asistenta* u TV aplikaciju;
- Izvršavanje glasovnih naredbi korisnika unutar TV aplikacije.

Na slici 3.1 je predstavljen dijagram sa toka izvršavanja programa, koji obuhvata gore pomenute module.

Slika 3.1 - Dijagram funkcionisanja TV aplikacije u kojoj je integrisan *Google asistent*

3.1.1 Obučavanje *Google Asistenta*

Kao što je već pomenuto, moguće je napraviti akcije za gotovo bilo kakvu akciju koja će biti proširenje GA, u zavisnosti od potrebe korisničke aplikacije. U okviru zadatka koji je obrađen u ovom radu je potrebno da se naprave sledeće akcije:

AKCIJA	OPIS AKCIJE
Channel up	Promena kanala na sledeći iz liste kanala
Channel down	Promena kanala na prethodni iz liste kanala
Volume up	Pojačavanje jačine tona
Volume down	Smanjivanje jačine tona
Mute	Potpuno utišavanje tona
Unmute	Vraćanje na pređašnju jačinu tona
Tune to channel (number)	Prebacivanje na kanal određenog broja

Tabela 1 – Spisak akcija za obučavanje GA

Kao ulazna tačka u zadatku, od korisnika se očekuje da izgovara punu rečenicu na engleskom jeziku, što može dodatno otežati prepoznavanje glasa zbog različitih akcenata i govornih dijalekata. Potrebno je pretpostaviti šta bi korisnik mogao reći umesto očekivane komande, kao i navesti što veći broj sinonima za reči kanal (eng. *Channel*), gore (eng. *Up*) i dole (eng. *Down*). Od velike važnosti je obučiti asistenta rečima koje se slično izgovaraju sa ključnim rečima poput para *channel – travel*, jer zbog loših uslova i šumova postoji mogućnost da GA ne „čuje“ reč kako je izgovorena.

3.1.2 Integracija *Google* Asistenta u TV aplikaciju

Kada je GA dobro obučen i akcije sa svojim namerama (eng. *Intents*) i TV aplikacija obezbeđuje osnovne funkcionalnosti potrebne za uspešno izvršenje ovog zadatka, ostaje treći i poslednji problem, njihovo povezivanje i integracija u jednu celinu.

Najpre je potrebno omogućiti snimanje zvuka kako bi GA mogao da snimi i obradi korisnikovu poruku. Takođe je neophodno aplikaciji omogućiti pristup Internetu, jer je GA virtuelni asistent u oblaku i neophodan mu je Internet kako zbog komunikacije sa posluživačem tako i zbog moguće interakcije sa korisnikom koji može da postavi pitanje ili zada naredbu za koju je potrebna pretraga Interneta.

Da bi se povezale gore pomenute dve celine potrebno je i proširiti TV aplikaciju određenim metodama koje će služiti za oslušivanje i primanje odgovora od GA, parsiranje istih i izvršavanje korisničkih naredbi. Implementacija istih detaljnije je opisana u narednom poglavlju.

3.1.3 Izvršavanje glasovnih naredbi korisnika unutar TV aplikacije

Ulazna tačka u aplikaciji, gde je smeštena sva logika programa je glavna aktivnost (eng. *Main Activity*) dok sve ostale aktivnosti i klase služe kao pomoćne. U njoj se nalaze sve metode i objekti neophodni za izvršavanje osnovnih funkcionalnosti. Za rad sa kanalima dovoljno je preuzeti sve dostupne kanale i manipulirati njima intuitivnom metodom *tuneToChannel()*. Za manipulaciju zvukom dovoljno je u projekat uključiti *AudioManager* koji sa svojim metodama omogućava izvršavanja željenih funkcionalnosti.

Važna metoda za ovu aplikaciju je i *onKeyDown()*. Ona „osluškuje“ događaje i reaguje na pritisak tastera. Potrebno ju je implementirati u aplikaciji kako bi se prilikom pritiska na dugme za glasovnu pretragu (eng. *voice search*) na daljinskom upravljaču pokrenuo *Google Asistent* i započeo interakciju sa korisnikom.

4. Programsko rešenje

U ovom poglavlju će biti detaljno opisano programsko rešenje integracije *Google Asistenta* u *Live TV* aplikaciju. TV Android aplikacija koja koristi funkcionalnosti programske podrške televizijskog prijemnika se razvija korišćenjem skupa alata i programskih elemenata za razvoj programske podrške u Android okruženju (eng. *Android Software Development Kit* – Android SDK). Kao što je i pomenuto u ranijim poglavljima, aplikativni sloj Androida naleže na sve ostale niže slojeve pa će sve korišćene funkcije i metode biti opisane.

4.1 Spisak modula

Android aplikacija je sačinjena od zasebnih programskih modula, čija je uloga prikaz grafičkih komponenti, prikupljanje komandi od strane daljinskog upravljača i modula koji predstavlja upravljačku logiku aplikacije. Upravljačka logika predstavlja programsku realizaciju načina na koji će određeni događaji biti prosleđeni programskoj podršci televizijskog prijemnika, kao i načine na koje će informacije iz programske podrške televizijskog prijemnika biti prikazane krajnjem korisniku putem različitih grafičkih komponenti aplikacije.

Prikazani su moduli ovih rešenja kao i fajlovi koji opisuju svaki modul pojedinačno (Tabela 2).

MODUL	KLASA
Modul za obučavanje GA	<i>Actions on Google (AI.API)</i>
Modul za integraciju GA u TV aplikaciju	MainActivity,AIListener
Modul za izvršavanje glasovnih komandi	MainActivity,AIListener

Tabela 2 – Moduli programskog rešenja

4.1.1 Modul za obučavanje GA

Temeljno obučavanje GA je od esencijalnog značaja za implementaciju ovog sistema. Napravljene su četiri namere (eng. *Intents*) što se može videti sa slike 4.1:

1. Namera dobrodošlice – za pokretanje rada GA;
2. Namera „izvrši akciju“ (*Perform action*) – za promenu kanala i jačine zvuka; na gore/dole i utišavanje i vraćanje na stanje pre utišavanja
3. Namera prebacivanja na određeni kanal (*Tune to*);
4. Namera za kraj akcije – suštinski nije od značaja.

Intents



Slika 4.1 – Namere za kreiranje akcija

Moguće je i smanjiti broj namera ali radi veće preciznosti razumevanja govora od strane asistenta kao i mogućnosti za boljom obukom istog, procenjeno je da je bolje raščlaniti namere prebacivanja na određeni kanal i nameru koja podržava sve ostale akcije. Za svaki od ovih namera, od kojih su gore pomenute dve najbitnije, su dodate komande koje se očekuju od korisnika kao i reči koje se slično izgovaraju ili homonimi što je ilustrovano na slikama 4.2 i 4.3.

Input	Output
tunes to channel	tunes to channel, want to Channel, cute animals, super Channel
tunes for channel	tunes for channel

[Click here to edit entry](#)

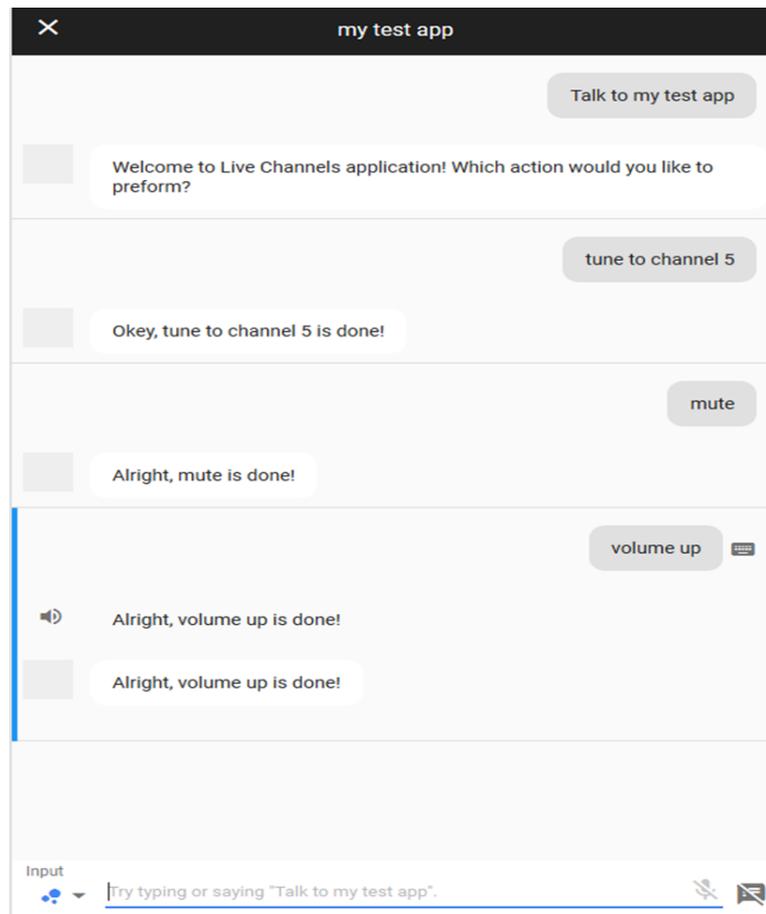
Slika 4.2 - Očekivan unos korisnika prilikom komande „prebaci na željeni kanal“

volume up	volume up, volume
channel up	channel up, channel app, travel up, tours travel, turn off off, turn off
mute	mute
volume down	volume down, volume, volume stops
channel down	channel down, tool stop, girl boss
unmute	unmute

Slika 4.3 - Očekivan unos korisnika prilikom ostalih ključnih komandi

Nakon obučavanja asistenta šta može da očekuje od korisnika potrebno je obučiti ga da daje određeni odgovor, koji će biti dovoljno precizan kako bi aplikacija „znala“ koju akciju treba da izvrši. Jedna od mogućnosti bila je da asistent daje predefinisani odgovor ali u toj formi on ne bi bio od koristi za aplikaciju, stoga je bilo potrebno dopustiti mu da ima pristup Internetu, odnosno posluživaču. Tada ga je programski moguće obučiti da daje odgovore u zavisnosti od parametara, koji su zapravo ključne reči poput: *channel up/down*, *mute/unmute* itd. Za te potrebe je korišćen programski jezik *JavaScript*.

U slučaju prepoznavanja glasovne komande, asistent će dati odgovor u formatu: „Alright/Okey **desired action** is done!“ ili će zamoliti sagovornika da ponovi komandu. Koristeći već pomenute Akcije na Guglu, izuzetno je bilo lako testirati rad asistenta kroz rad u konzoli, o čemu svedoči i slika 4.4. Na slici je prikazan primer dijaloga prilikom testiranja *Google Asistenta* nakon obučavanja.



Slika 4.4 – Testiranje rada GA

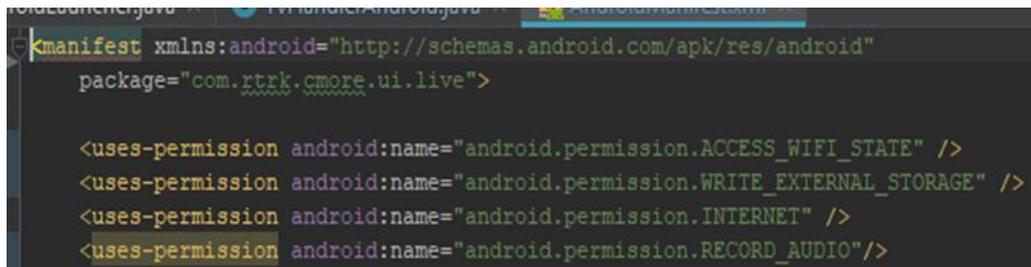
Na ovaj način moguće je proveriti da li je obučavanje *Google Asistenta* uspešno izvršeno. Umnogome je olakšan prihvatanje odgovora i njegova obrada od strane aplikacije zahvaljujući ovom formatu odgovora asistenta. Takođe, ne treba zaboraviti da aplikacija prihvata odgovor u vidu JSON objekta a ne u vidu tekstualnog zapisa, kako je predstavljeno na slici.

4.1.2 Modul za integraciju GA u TV aplikaciju

Nakon uspešno završene obuke *Google Asistenta*, prelazi se na integraciju u okviru TV aplikacije što će biti opisano u ovom poglavlju.

Za potrebe prepoznavanja govora ili teksta je korišćen ugrađen mehanizam kog poseduje *Google Asistent*. Treba napomenuti da je moguće koristiti i sopstveni mehanizam za prepoznavanje govora ali se u tom slučaju sama implementacija i integracija u TV aplikaciju znatno razlikuju.

Najpre je potrebno izmeniti fajl *AndroidManifest.xml*. U tom fajlu se deklarise ime aplikacije, sve njene komponente uključujući sve aktivnosti (eng. *Activities*), servise (eng. *Services*), obezbeđivače sadržaja (eng. *Content Providers*), dodeljuju dozvole pristupa aplikaciji i dr. Neophodno je dozvoliti snimanje audio sadržaja i pristup Internetu što je prikazano na slici 4.5.



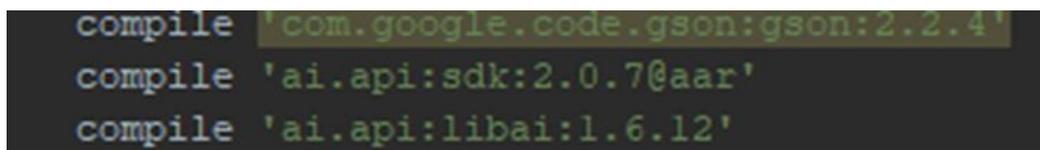
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.rtrk.more.ui.live">

    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
```

Slika 4.5 - Deo *AndroidManifest.xml* datoteke

Pre implementacije neophodnih metoda za integraciju sa asistentom, potrebno je još dopuniti fajl *build.gradle* sa sledećim dinamičkim bibliotekama (slika 4.6):

- Sdk 2.0.7 – dinamička biblioteka koja dozvoljava korišćenje glasovnih komandi i dijalog sa asistentom
- Libai:1.6.12 – dinamička biblioteka koja ima gotovo identičnu funkcionalost kao prethodna
- Gson 2.2.4 – dinamička biblioteka koja omogućava korišćenje JSON objekata



```
compile 'com.google.code.gson:gson:2.2.4'
compile 'ai.api:sdk:2.0.7@aar'
compile 'ai.api:libai:1.6.12'
```

Slika 4.6 – Neophodne dinamičke biblioteke

Nakon uspešno izvršenih prethodnih koraka preostaje da se implementira programski deo rešenja. Glavna aktivnost mora da implementira spregu *AIListener* sa svojim metodama:

void onResult(AIResponse result)	Služi za obradu odgovora pristiglog od strane <i>Google Asistenta</i> .
void onError(AIError error)	Služi za obradu greške ukoliko do nje dođe prilikom komunikacije sa GA.
void onAudioLevel(float error)	<i>Callback</i> metoda za vizuelizaciju jačine zvuka.
void onListening Started()	Metoda za indicaciju početka slušanja korisničkog unosa.
void onListeningFinished()	Metoda za indicaciju kraja slušanja korisničkog unosa.
void onListeningCanceled()	Metoda za indicaciju prekida slušanja korisničkog unosa.

U metodi *onCreate()* se vrši konfiguracija *Google Asistenta*, pomoću objekta tipa *AIConfiguration*. Potrebno je da se podesi jedinstveni kod URI napravljenih akcija, podržani jezik i sistem za prepoznavanje. Objekat tipa *AIService* preuzima konfiguraciju i kreira oslušivač (eng. *listener*) za korisničke komande. To je prikazano na slici 4.7.

```
final AIConfiguration config = new AIConfiguration(clientAccessToken,
    AIConfiguration.SupportedLanguages.English,
    AIConfiguration.RecognitionEngine.System);

aiService = AIService.getService(context: this, config);
aiService.setListener(this);
```

Slika 4.7 - Konfiguracija asistenta sa aplikacijom

Kreiran je objekat *config* tipa *AIConfiguration* koji sadrži URI akcija, engleski jezik kao jezik prepoznavanja i sistem za prepoznavanje govora. Nakon toga *aiService* kreira oslušivač.

Pošto je oslušivač napravljen, postavlja se pitanje kada treba da počne da „sluša“. Da ne bi bespotrebno „slušao“ i trošio resurse, poziva se metoda *onKeyDown()*, te se pritiskom na taster na daljinskom upravljaču aktivira *AIService* i započinje slušanje korisničkih komandi. Implementacija te metode je data na slici 4.8.

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (event.getAction() == KeyEvent.ACTION_DOWN) {
        if (keyCode == 231 || keyCode == 84) {
            aiService.startListening();
            return true;
        }
    }
}
```

Slika 4.8 – *onKeyDown* metod

Slušanje govornog unosa je promenljive dužine u zavisnosti od preciznosti prepoznavanja istog, spoljnog uticaja i šumova, kao i brzine izgovaranja reči samog korisnika.

4.1.3 Modul za izvršavanje glasovnih komandi

Sledi pojašnjenje svake od gore navedenih metoda , neophodnih za implementaciju ovog modula.

4.1.3.1 void *onResult*(AIResponse result)

Metoda *onResult* je od suštinskog značaja jer se u njoj dobavljaju odgovori od strane *Google* Asistenta. Odgovor stiže u obliku objekta u JSON formatu. Dobavljeni podaci se smeštaju u strukturu podataka *Map*, koja sadrži parove tekstualni objekat – JSON objekat. Ključ u mapi predstavlja jednu od dve namere (*perform_action* ili *tune_to*) dok vrednost koja je vezana za ključ predstavlja korisnički zahtev (npr. *Channel up*). Od krucijalnog značaja za aplikaciju je dobijanje te vrednosti vezane za ključ kako bi se izvršila željena komanda. Raščlanjivanjem (parsiranjem) JSON objekta i smeštanjem podataka iz njega u tekstualni objekat se dolazi do mogućnosti za manipulaciju primljenim podacima. Ukoliko se uspešno raščlani asistentov odgovor moguće je dobiti zahtev korisnika, akciju koju treba izvršiti kao i druge parametre. Obrada podataka u ovoj metodi izvršava se u posebnoj niti (eng. *thread*). Za parsiranje parametara je koršćena pomoćna metoda *parseRequest*, prikazana na slici 4.9.

```

String parseRequest(Result result){

    // Get parameters

    if (result.getParameters() != null && !result.getParameters().isEmpty())
    {
        for (final Map.Entry<String, JsonElement> entry : result.getParameters().entrySet())
        {
            parameterString += "(" + entry.getKey() + ", " + entry.getValue() + ") ";
            queryFromAssistant = entry.getValue().toString();
        }
    }

    Log.d(LOG_TAG, msg: "Query:" + result.getResolvedQuery() +
        "\nAction: " + result.getAction() +
        "\nParameters: " + parameterString);

    /* parse got query to make an action */
    if(queryFromAssistant.length() > 0) {
        recognized = queryFromAssistant.substring(1, queryFromAssistant.length() - 1);
    }

    return recognized;
}

```

Slika 4.9 – Parsiranje parametara

Prilikom govornog unosa korisnika, šalje se JSON zahtev prema asistentu i očekuje se odgovor od njega. Taj odgovor je smešten u objekat tipa *Result*, te je potrebno preuzeti podatke iz tog objekta, smestiti ih u programsku strukturu mape. Nakon toga se podaci smeštaju u tekstualni tip objekta (*String*), koji je vrlo pogodan za manipulaciju, i njegovim internim metodama se dobija krajnja vrednost, odnosno korisnički zahtev.

Nakon sto se prispeli odgovor uspešno obradi i prepozna pristigla komanda, kao rezultat prethodno prikazane *parseRequest* metode, pristupa se izvršavanju komande. Tip komande se prepoznaje u okviru *switch-case* iskaza/naredbe. Postoje više slučajeva korišćenja (eng. *Use case*) u zavisnosti od zahteva korisnika. U slučaju zahteva za promenom kanala (*channel up/down*), potrebno je iterirati kroz listu dostupnih kanala povećavanjem/smanjivanjem indeksa elementa u listi za jedan. Kanali se dobavljaju metodom *getBrowsableChannelList ()*. Ako korisnik želi da prebaci na određeni kanal onda se koristi metoda *tuneToChannel ()* gde se kao parametar prosleđuje broj kanala iz liste. Ukoliko se prosledi broj kanala koji ne postoji u listi, korisniku se ispisuje poruka o grešci da traženi kanal ne postoji. Na slici 4.10 se nalazi implementacija rešenja za slučaj prebacivanja kanala unapred. Slučaj prebacivanja kanala unazad je gotovo identičan, jedino se indeks kanala smanjuje a ne povećava, kao u prethodnom slučaju.

```

switch (recognized) {
    case "channel up":
        System.out.println("@@@ CHANNEL UP ZAHTEV");
        channelList = getBrowsableChannelList();
        tuneToChannel((Integer) channelList.get(ind % numOfChannels));
        ind++;
}

```

Slika 4.10 - Slučaj korišćenja *channel up*

Za manipulaciju jačinom tona, korisnik zadaje komande *volume up/down* ili *mute/unmute*. Ukoliko je slučaj gde se zvuk pojačava ili stišava, potrebno je koristiti metode objekta *AudioManager*. Najbitnije metode su *getStreamMaxVolume ()*, koja predstavlja referentnu vrednost maksimalne jačine tona, i *setStreamVolume ()*, pomoću koje se jačina tona podešava na željenu vrednost, koja se prosleđuje kao parametar. Na slici 4.12 je prikazana implementacija programskog rešenja za menjanje jačine tona. Najpre je potrebno da se dobavi maksimalna moguća jačina tona, koja služi kao referentna vrednost za trenutnu jačinu. Ako je moguće, vrednost trenutne jačine se poveća ili smanji i prosledi metodi *setStreamVolume* objekta tipa *AudioManager*, koji je zadužen za izmenu jačine tona.

```

case "volume up":
    Toast.makeText(context: AndroidLauncher.this, text: "Volume up", Toast.LENGTH_LONG).show();
    max = mAudioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
    currentVolume++;
    if (currentVolume > max)
        currentVolume = (int) max;

    mAudioManager.setStreamVolume(AudioManager.STREAM_MUSIC, currentVolume, flags: 0);
    break;

```

Slika 4.11 - Slučaj korišćenja pojačavanja tona

U slučaju potpunog utišavanja tona, koriste se iste metode, jedina razlika se ugleda u tome što se kao parametar jačine tona prosleđuje vrednost 0. Da bi se uspešno vratila jačina zvuka koja je bila i pre korisničkog zahteva *mute*, neophodno je zapamtiti vrednost jačine zvuka pre tog zahteva i nakon naredbe *unmute* ta vrednost se prosleđuje metodi *setStreamVolume ()*. Ukoliko korisnik pritisne zada naredbu *mute/unmute* pa ponovo zada istu naredbu, dobiće poruku da je njegova naredba već izvršena. Slika 4.12 prikazuje implementaciju za ovog slučaja korišćenja.

```

case "mute":
    if (!isMuted) {
        Toast.makeText(context: AndroidLauncher.this, text: "Muted", Toast.LENGTH_LONG).show();
        rememberVolume = currentVolume;
        currentVolume = 0;
        mAudioManager.setStreamVolume(AudioManager.STREAM_MUSIC, currentVolume, flags: 0);
        isMuted = true;
    } else {
        Toast.makeText(context: AndroidLauncher.this, text: "Already muted", Toast.LENGTH_LONG).show();
    }
    break;

```

Slika 4.12 - Slučaj korišćenja potpunog utišavanja tona

Slučaj korišćenja gde korisnik zadaje tačan broj kanala na koji želi da se prebaci je specifičan pre svega što iziskuje od korisnika da pravilo izgovori više reči, a sa druge strane otežava programeru kako obuku asistenta, tako i raščlanjivanje parametara koje dobije od istog. Iz tog razloga su implementirane dve pomoćne metode *parseTuneToChannel* i *isInteger*, koje proveravaju da li se među pristiglim parametrima nalazi broj kanala i ako se nalazi, taj broj se dobavlja aplikaciji.

Ukoliko je korisnik zadao naredbu prebacivanja na tačno određeni kanal, onda je potrebno dobiti informacije i o broju kanala na koji želi da se prebaci. S obzirom na činjenicu na broj može da se napiše i u obliku teksta, potrebna je dodatna obrada pristiglog rezultata za slučaj da nije predstavljen u obliku teksta već numerički. Iz tog razloga je implementirana i koristi se pomoćna funkcija *isInteger*. Implementacija ovog slučaja korišćenja prikazana je na slici 4.13. Vrednosti parametara se proveravaju i, u slučaju zahteva sa indeksom kanala koji ne postoji u listi, korisniku se prikazuje *toast* poruka obaveštenja da je izabrao nepostojeći kanal.

```

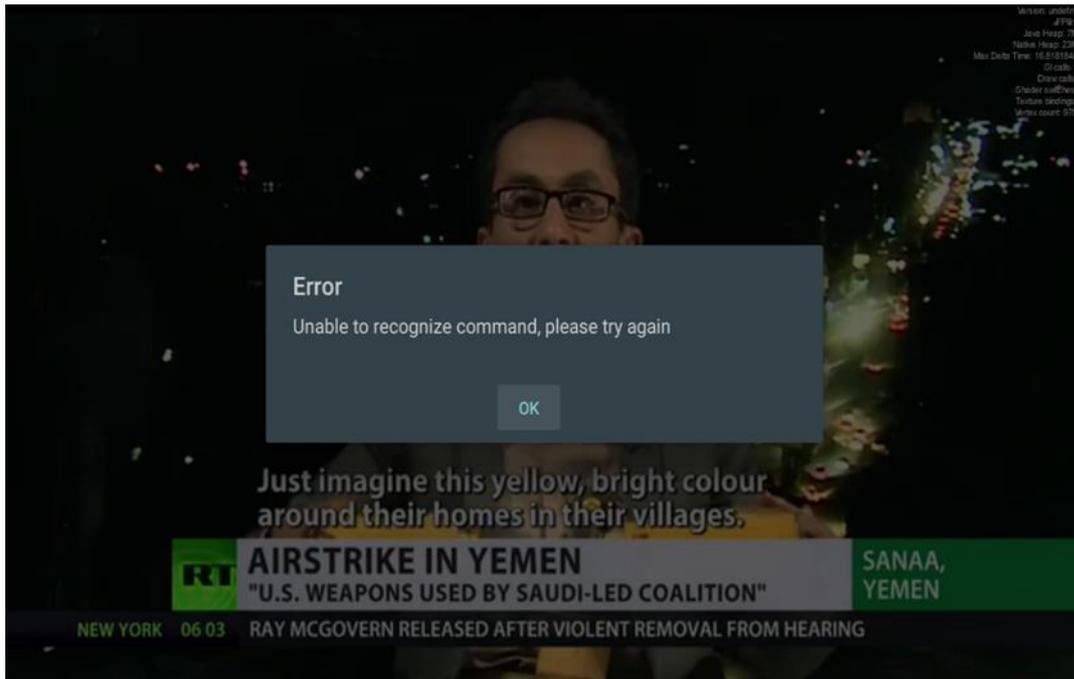
case "tune to channel":
    Log.d(LOG_TAG, msg: "TUNE TO CHANNEL");
    parseTuneToChannel(result);
    if (isInteger(numberInQuery)) {
        desiredChannel = Integer.parseInt(numberInQuery);
        // change channel only if channels are existing in list
        if (desiredChannel > 0 && desiredChannel < 27){
            tuneToChannel(desiredChannel);
        }else {
            Toast.makeText( context: AndroidLauncher.this, text: "This channel does not exist, please try again.", Toast.LENGTH_LONG).show();
        }
    } else if (isInteger(numberInParameter)) {
        desiredChannel = Integer.parseInt(numberInParameter);
        if (desiredChannel > 0 && desiredChannel < 27){
            tuneToChannel(desiredChannel);
        }else {
            Toast.makeText( context: AndroidLauncher.this, text: "This channel does not exist, please try again.", Toast.LENGTH_LONG).show();
        }
    }
    break;
default:

```

Slika 4.13 - Slučaj korišćenja *tuneToChannel*

4.1.3.2 void onError(AIError error)

Kao što je u tekstu iznad pomenuto, postoji mogućnost da je korisnički unos nejasan, da uopšte nije zadat, da je nivo šuma previsok ili da je došlo do neke greške prilikom komunikacije sa asistentom, i tada se u posebnoj niti izvršava ispis poruke o grešci. Na ekranu se pojavljuje prozor sa porukom obaveštenja korisniku (*Alert Dialog*) (slika 4.14) i molbom za ponovnim pokušajem.



Slika 4.14 - Izgleda ekrana sa porukom o grešci

4.1.3.3 void onListeningFinished()

Ova metoda predstavlja indicaciju da je slušanje korisničke naredbe završeno i služi kao pomoć prilikom razvijanja i testiranja aplikacije.

4.1.3.4 void OnListeningCanceled()

Slično kao i prethodna metoda, služi da obavesti programera da je slušanje komande prekinuto iz nekog razloga.

Metode *onAudioLevel (float level)* i *onListeningStarted ()* su definisane jer su obavezan deo sprege *AllListener* ali ništa u njima nije implementirano.

5. Rezultati

U okviru ovog poglavlja prikazani su rezultati testiranja i verifikacije TV aplikacije. Testiranje je vršeno ručno u Guglovoj *Live Channels* aplikaciji. U cilju potvrđivanja funkcionalnosti realizovanog rešenja, sprovedeno je nekoliko ispitivanja:

- Ispitivanje uspešnosti svakog od slučaja korišćenja;
- Ispitivanje vremena odziva *Google Asistenta*;
- Ispitivanje uspešnosti prepoznavanja komandi od strane asistenta.

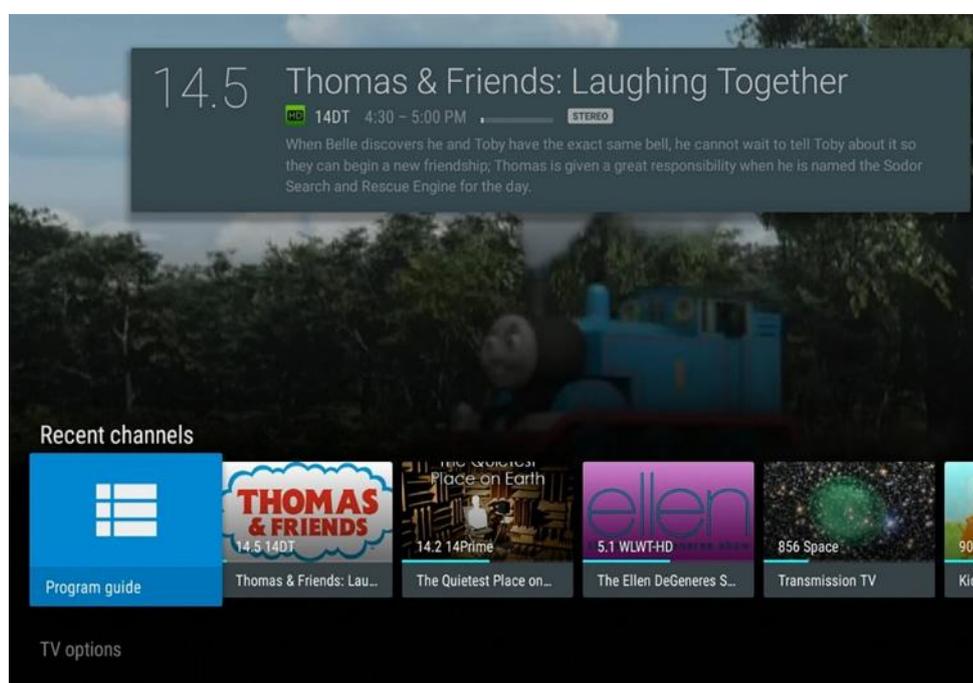
5.1 Ispitivanje uspešnosti svakog od slučaja korišćenja

U okviru ovog testiranja ispitano je ponašanje aplikacije prilikom zadavanja osnovnih komandi pri normalnim uslovima. Zbog nemogućnosti automatizacije procesa testiranja, kao i zbog mogućnosti promene spoljašnjih uticaja, vršeno je ručno testiranje na engleskom jeziku, ponovljeno deset puta. Rezultati su prikazani u Tabeli 3 predstavljaju verifikaciju funkcionalnosti, dok su u poglavlju 5.3 dati rezultati o preciznosti prepoznavanja komandi.

Ispitni slučaj	Rezultat
<i>Channel up</i>	USPEŠNO ZAVRŠEN
<i>Channel down</i>	USPEŠNO ZAVRŠEN
<i>Volume up</i>	USPEŠNO ZAVRŠEN
<i>Volume down</i>	USPEŠNO ZAVRŠEN
<i>Mute</i>	USPEŠNO ZAVRŠEN
<i>Unmute</i>	USPEŠNO ZAVRŠEN
<i>Tune to channel</i>	USPEŠNO ZAVRŠEN

Tabela 3 - Rezultati uspešnosti izvršavanja naredbi

Naredne tri slike ilustruju ponašanje aplikacije prilikom promene kanala, pojačavanja i potpunog utišavanja tona (slike 5.1, 5.2, 5.3).



Slika 5.1 – Izgled aplikacije prilikom promene kanala



Slika 5.2 – Grafički prikaz na ekranu prilikom pojačavanja tona



Slika 5.3 - Grafički prikaz na ekranu prilikom potpunog utišavanja tona

5.2 Ispitivanje vremena odziva Google Asistenta

U ovom delu testiranja izvršeno je merenje vremena između glasovnog unosa korisnika i uzvraćenog odgovora od strane *Google Asistenta*. Potrebno je uzeti u obzir kompleksnost komandi koja se razlikuje od slučaja do slučaja (npr. Jednostavna komanda – *mute*, složena komanda – *tune to channel five*). Razgovetnost glasovnog unosa je takođe bitan faktor koji treba uzeti u obzir prilikom merenja vremena odziva. Izvršeno je po tri merenja za svako od sedam mogućih komandi. Dobijeni rezultati prikazani su u Tabeli 4.

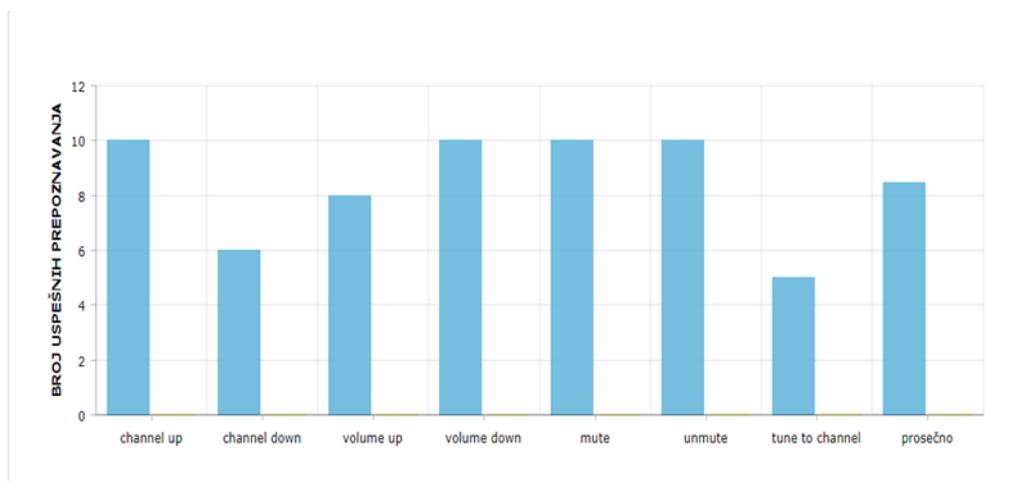
Komanda	Merenje 1	Merenje 2	Merenje 3	Prosek
Channel up	2,937 sec	2,696 sec	2,434 sec	~2,689 sec
Channel down	3,660 sec	2,632 sec	2,301 sec	~2,864 sec
Volume up	3,815 sec	2,711 sec	2,574 sec	~3,033 sec
Volume down	2,376 sec	2,855 sec	2,843 sec	~2,688 sec
Mute	2,632 sec	3,818 sec	2,305 sec	~2,918 sec
Unmute	2,602 sec	2,446 sec	3,124 sec	~2,724 sec
Tune to channel	3,392 sec	3,606 sec	4,731 sec	~3,909 sec
Prosečno vreme odziva za sve komande				~2,975 sec

Tabela 4 - Rezultati merenja odziva za komande

Prosečno vreme čekanja na odgovor je između 2 i 3 sekunde, te se može zaključiti da li je ovo vreme prihvatljivo sa aspekta korisničkog iskustva ili nije. Uspešnost prepoznavanja komandi od strane *Google Asistenta* je 85%, ali će o tome detaljnije biti objašnjeno u narednom poglavlju.

5.3 Ispitivanje uspešnosti prepoznavanja komandi od strane asistenta

U ovom delu verifikacije izvršeno je testiranje uspešnosti prepoznavanja komandi od strane *Google Asistenta*, pri normalnim spoljašnjim faktorima kao što su raznorazni šumovi, pozadinski govor, saobraćaj koji se čuje kroz otvoreni prozor i drugo. Vršeno je po deset merenja za svaku od mogućih komandi a rezultati su predstavljeni na sledećem dijagramu na slici 5.4.



Slika 5.4 - Procenat uspešnosti prepoznavanja komandi

U odnosu na generalnu uspešnost prepoznavanja korisničkih komandi od strane *Google Asistenta* u drugim aplikacijama (oko 95%), uspešnost prepoznavanja u ovoj aplikaciji je nešto niža (80%). Tu razliku opravdava činjenica da je GA obučen za desetak različitih potencijalnih korisničkih unosa po komandi, za razliku od drugih aplikacija koje koriste ovaj servis, gde je broj mogućih varijacija glasovnih unosa za jednu komandu veći i nekoliko desetina puta. Drugim rečima, GA je u drugim aplikacijama mnogo bolje obučen.

Na osnovu izvršenih merenja uočava se odlično prepoznavanje komandi *channel up*, *volume down* i *mute/unmute*, dok je očekivan rezultat postignut za najzahtevniju komandu *tune to channel (number)*. Ona se sastoji iz dva parametra: akcije i broja kanala te je potrebno izvršiti kompleksniju obradu, ali i zadati kompleksniju naredbu. Naravno, treba uzeti u obzir i način izgovaranja zadatih komandi, te bi testiranje bilo verodostojnije da je u njemu učestvovalo više participanata. Dobijeni rezultati su zadovoljavajući, s' tim da postoji prostor za daljim razvijanjem i boljim obučavanjem *Google Asistenta*.

6. Zaključak

U ovom radu predstavljeno je jedno rešenje integracije *Google Asistenta* u Guglovu *Live Channels* aplikaciju. U tu svrhu bilo je neophodno obučiti GA, izvršiti njegovu integraciju u samu aplikaciju i na kraju obraditi pristigli odgovor od strane GA i izvršiti željenu komandu. S obzirom da se radi o aplikativnom sloju Androida, rešenje je implementirano u Java programskom jeziku uz minimalno korišćenje *JavaScript* programskog jezika za obuku GA.

Rezultati pokazuju da aplikacija podržava svih sedam navedenih funkcionalnosti (korisničkih komandi). Vreme proseka je dobro, i u proseku je potrebno oko tri sekunde od trenutka kada korisnik zada komandu do trenutka kada je komanda obrađena i izvršena. Takođe, ukoliko se obrati pažnja na pravilan izgovor, procenat prepoznatih komandi od strane GA srazmerno raste.

Rešenje integracije GA u TV aplikaciju je lako proširivo i drugim komandama poput prikazivanja elektronskog programskog vodiča (EPG) ili slike-u-slici (PIP). Neophodno je proširiti obuku asistenta za prepoznavanje dodatnih komandi i proširiti slučajeve korišćenja u aplikaciji, implementirajući funkcionalnosti dodatnih komandi.

Dalji rad na unapređivanju ovog rešenja mogao bi da bude usmeren u pravcu dodatne obuke GA i proširivanju dodatnih funkcionalnosti *Live Channels* aplikacije.

7. Literatura

- [1] Računarstvo u oblaku (*Cloud computing*), <https://www.zdnet.com/article/what-is-cloud-computing-everything-you-need-to-know-from-public-and-private-cloud-to-software-as-a/>, jun 2018
- [2] Dr Milan Z. Bjelica, dr Nikola Teslić, mr Velibor Mihić, Softver u televiziji i obradi slike 1, Novi Sad: FTN izdavaštvo, 2017
- [3] Istraživanje – televizija kao dominantan izvor zabave, <http://www.digitalcenter.org/web-insight-views-about-entertainment-sources-2010/>, 2010
- [4] Android TV Apps Development, Paul Trebilcox-Ruiz, ISBN: 978-1-4842-1784-9, 2016
- [5] TIF, <https://source.android.com/devices/tv/>, jun 2018
- [6] Google Assistant SDK , <https://developers.google.com/assistant/sdk/>, jun 2018
- [7] Elenora Nan, “Upravljanje pametnom kućom uz pomoć *Google* asistenta”, Univerzitet u Novom Sadu, FTN, 2017
- [8] Akcije na Guglu, <https://developers.google.com/actions/>, jun 2018
- [9] Google Live Channels App, <https://androidtv.news/2016/10/get-started-live-channels/>, jun 2018
- [10] M.Vidakovic, N.Teslic, T.Maruna, and V.Mihic: Android4TV: a proposition for integration of DTV in Android devices, IEEE 30th International Conference on Consumer Electronics (ICCE), Las Vegas, January 2012

-
- [11] Dr Ištvan Pap, Dr Nemanja Lukić, Projektovanje namenskih računarskih sistema 1, Novi Sad: FTN izdavaštvo 2016
- [12] C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatawy and D. Suendermann-Oeft, “Comparing Open-Source Speech Recognition Toolkits”, Technical Report of the Project OASIS, Germany, 2014.
- [13] Android SDK, <https://developer.android.com/studio/>, jun 2018
- [14] M. Robin and M. Poulin: “Digital television fundamentals – Design and Installation of Video and Audio Systems”, McGraw-Hill, 1997
- [15] J.Bloch: EffectiveJava (2nd edition), Addison-Weasly, 2008