УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ

Младен Илић

Једно решење хибридног уређаја за репродукцију дигиталног и Интернет телевизијског садржаја базираног на Андроид оперативном систему

МАСТЕР РАД

Нови Сад, јул 2015

УНИВЕРЗИТЕТ У НОВОМ САДУ ● **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:			
Идентификациони број, **ИБР**:			
Тип документације, **ТД**:		Монографска документација	
Тип записа, **ТЗ**:		Текстуални штампани материјал	
Врста рада, **ВР**:		Дипломски – мастер рад	
Аутор, **АУ**:		Младен Илић	
Ментор, **МН**:		др Илија Башичевић	
Наслов рада, **НР**:		Једно решење хибридног уређаја за репродукцију дигиталног и Интернет телевизијског садржаја базираног на Андроид оперативном систему	
Језик публикације, **ЈП**:		Српски / латиница	
Језик извода, **ЈИ**:		Српски	
Земља публиковања, **ЗП**:		Република Србија	
Уже географско подручје, **УГП**:		Војводина	
Година, **ГО**:		2015	
Издавач, **ИЗ**:		Ауторски репринт	
Место и адреса, **МА**:		Нови Сад; трг Доситеја Обрадовића 6	
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)		7/40/0/0/15/0/0	
Научна област, **НО**:		Електротехника и рачунарство	
Научна дисциплина, **НД**:		Рачунарска техника	
Предметна одредница/Кqучне речи, **ПО**:		Дигитална телевизија, OTT, Андроид, електронски програмски водич, ТВ, ДТТ	
УДК			
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад	
Важна напомена, **ВН**:			
Извод, **ИЗ**:		У овом раду је реализована интеграција програмске подршке за дигиталну телевизију на хибридном СТБ (енгл. set top box) уређају проширењем Андроид оперативног система. Приказано је претраживање сервиса дигиталне телевизије и сервиса који се налазе на мрежном послужиоцу, добављање електронског програмског водича као и комуникација са мрежним послужиоцем Интернет мреже користећи OTT (енгл. over the top) начин комуникације. У раду су приказани дизајн софтвера и процедуре за верификацију које су примењене.	
Датум прихватања теме, **ДП**:			
Датум одбране, **ДО**:			
Чланови комисије, **КО**:	Председник:	Др Иван Мезеи, доцент	
	Члан:	Др Иштван Пап, доцент	Потпис ментора
	Члан, ментор:	Др Илија Башичевић, ванред.проф.	

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**:		Master Thesis	
Author, **AU**:		Mladen Ilic	
Mentor, **MN**:		PhD Ilija Basicevic	
Title, **TI**:		One solution of Android powered hybrid STB device for reproduction of DVB and OTT content	
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**:		2015	
Publisher, **PB**:		Author's reprint	
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6	
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		7/40/0/0/15/0/0	
Scientific field, **SF**:		Electrical Engineering	
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, **S/KW**:		Digital television, OTT, Android, EPG, TV, DTT	
UC			
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, **N**:			
Abstract, **AB**:		This paper describes integration of software stack for digital television on Android OS based hybrid set top box (STB) device. The following scenarios are presented: service installation for digital terrestrial television (DTT) and Over the top (OTT) services, electronic program guide and communication between STB device and OTT server. The paper presents the software design and verification procedures that were used.	
Accepted by the Scientific Board on, **ASB**:			
Defended on, **DE**:			
Defended Board, **DB**:	President:	PhD Ivan Mezei	
	Member:	PhD Istvan Pap	Menthor's sign
	Member, Mentor:	PhD Ilija Basicevic	

SADRŽAJ

SPISAK SLIKA

SKRAĆENICE

DTT - *Digital terrestrial television*, digitalna zemaljska televizija

OTT - *Over The Top*, način komunikacije sa internet mrežnim poslužiocem

PAL - *Phase Alternating Line,* standard za analognu televiziju

NTSC - *National Televison System Committee*, Nacionalni komitet za televizijski sistem

VOD - *Video on demand*, Video na zahtev

PVR - *Personal video recording*, snimač video sadržaja

STB - *Set top box,* uređaj za reprodukciju televizijskog sadržaja

SGL - *Scalable Graphics Library*, biblioteka za skaliranje grafike

EPG - *Electronic program guide*, elektronski programski vodič

LCN - *Logical channel number*, logički broj kanala

API - *Application Programming Interface*, aplikacioni programski interfejs

UML - *Unified Modeling Language*, objedinjeni jezik za modelovanje

MAL - *Middleware Abstraction Layer*, sloj programske podrške koji apstrakuje srednji sloj

1. Uvod

U ovom radu je predstavljena realizacija proširenja Android operativnog sistema integracijom postojeće programske podrške za digitalnu televiziju. Prikazana je realizacija tri sloja programske podrške i objašnjena komunikacija između njih pomoću više različitih mehanizama prenosa podataka.

Cilj ove implementacije je da omogući pretragu i prikaz servisa digitalne televizije kao i servisa koji se nalaze na mrežnom poslužiocu. Prikazano je i prikupljanje elekrtronskog programskog vodiča za mrežne i servise digitalne televizije i propagacija tog sadržaja do grafičke korisničke sprege.

U ovom rešenju je opisana komunikacija sa mrežnim poslužiocem putem Internet mreže koristeći OTT (engl. Over The Top) način komunikacije.

Ovo rešenje se može koristiti na više različitih platformi, čiji srednji sloj (engl. middleware) je kompatabilan sa definisanim aplikacionim programskim interfejsom (engl. Application Programming Interface - API) u MAL (engl. Middleware Abstraction Layer) sloju programske podrške. Takođe ovo rešenje je nezavisno i od verzije Android operativnog sistema.

Skupovima testnih slučajeva je utvrđeno identično, ispravno ponašanje, čime je potvrđena ispravnost datog rešenja.

Ovaj rad je sačinjen od sedam poglavlja. U drugom poglavlju dat je opis Android operativnog sistema, razvoj digitalne televizije kao i opis OTT sadržaja na mrežnom poslužiocu. U trećem poglavlju opisani su slojevi programske podrške kao i komunikacija između njih. Takođe u njemu se nalazi koncept pretrage servisa i koncept dobavljanja elektronskog programskog vodiča. Četvrto poglavlje sadrži opis realizacije slojeva programske podrške, pretrage digitalnih i mrežnih servisa i dobavljanje elektronskog programskog vodiča. U petom

poglavlju dat je opis procesa verifikacije i testiranja rešenja kao i dobijeni rezultati. Šesto poglavlje sadrži kratak pregled onoga što je urađeno u ovom radu i kakvi su dalji pravci razvoja. U poslednjem sedmom poglavlju dat je spisak literature koja je korišćena za izradu ovog rada.

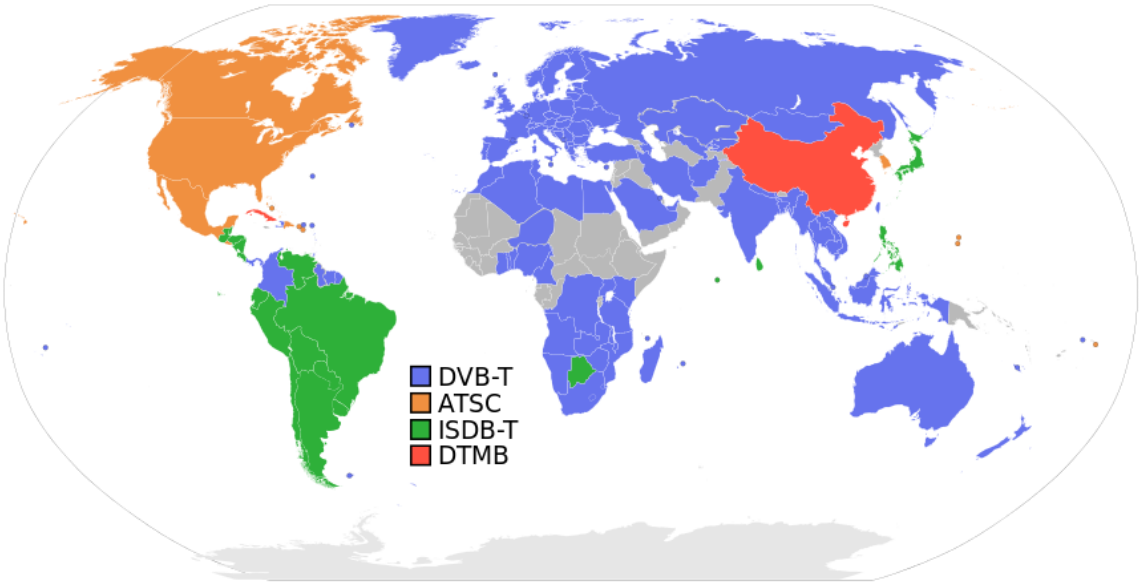
2. Teorijske osnove

U ovom poglavlju je opisan razvoj digitalne televizije, pregled njenih standarda, sprega internet sadržaja i digitalne televizije kao i opis OTT tehnologije prenosa internet sadržaja na digitalni televizijski prijemnik. Takođe, u nastavku je dat kratak opis Android platforme i njegovih komponenti.

2.1 Razvoj digitalne televizije

Digititalna televizija je nastala kao potreba da se zamene postojeći standardi prenosa televizijskog signala analognim putem. Standardi za emitovanje analogne televizije postoje vise od pedeset godina. Druga verzija NTSC standarda (bila je u upotrebi u većini zemalja Amerike, Južnoj Koreji, Japanu i dr.) 1953.godine je usvojena i omogućila je emitovanje televizijske slike u boji. U Evropi 60-tih godina prošlog veka se razvijao PAL standard za emitovanje televizijskog signala koji je u upotrebi od 1967.godine sirom sveta [1]. Najveći problem analogne televizije je povećanje smetnji na signalu koji se emituje usled spoljašnjih uticaja. Kao posledica napredovanja tehnologije snimanja sadržaja, pojavila se potreba za prenosom televizijskog sadržaja većeg kvaliteta što je dovelo do potrebe uvođenja novih načina prenosa, definisanja novih standarda usled čega je nastala digitalna televizija u obliku koji danas poznajemo.

Osnovna karakteristika digitalne televizije predstavlja prenos zvuka i slike sa dodatnim informacijama u digitalnom formatu [2]. Digitalni prenos obezbeđuje bolji kvalitet slike i zvuka koji više ne mogu biti u prenosu ometani interferencijom sa drugim signalima bez obzira na rastojanje na koje se slika i zvuk prenose. Slika i zvuk koju digitalni signal nosi su isti kao i na izvoru emitovanja, sve dok signal ne postane toliko slab da prijem više nije moguć.



Slika 1: Pregled standarda digitalne televizije po zemljama sveta

Kao i u analognoj televiziji i u digitalnoj postoji više standarda za prenos televizijskog signala. Trenutno najrasprostranjeniji standardi digitalne televizije su:

- ATSC (eng. *Advanced Television System Committee*) [3] standardi koriste se za prenos zemaljskog, kablovskog i satelitskog signala u SAD-u, Kanadi, Južnoj Koreji i dr. i prirodni je naslednik NTSC standarda analogne televizije.
- DVB (eng. *Digital Video Broadcasting*) standardi se najviše koriste u Evropi u zemljama koje su nekad koristile PAL i SECAM standarde analogne televizije. Postoji više tipova DVB standarda i to:
  - DVB-T/T2 (eng. *Terrestrial*) – definiše zemaljski prenos digitalnog televizijskog signala
  - DVB-C/C2 (eng. *Cable*) – definiše prenos signala putem kablovske mreže
  - DVB-S/S2 (eng. *Satellite*) – definiše satelitski prenos digitalnog televizijskog signala
  - DVB-H (eng. *Handheld*) – definiše prenos digitalnog televiziskog signala za mobilne uređaje
- DTMB (eng. *Digital Terrestrial Multimedia Broadcast*) – koristi se u Kini, Hokgkongu i Makau i definiše standard za prenos zemaljskog signala ka fiksnim i mobilnim uređajima
- ISDB (eng. *Integrated Services Digital Broadcasting*) – standard koji definiše digitalnog radio i televizijskog signala i koristi u Japanu, Brazilu i drugim zemljama.

Pojavom digitalne televizije obezbeđuje se bolji kvalitet slike i zvuka koji su istog kvaliteta na izvoru emitovanja i na svim mestima gde je moguće primiti taj signal. Krajnji korisnik može birati format slike koji želi da gleda (4:3,16:9), tip zvuka (npr. jednokanalni, višekanalni), zvučne trake koje emiter emituje (npr. srpski,engleski ili nemački jezik), da ima uvid u emisije koje može da podgleda u budućnosti uz pomoć elektronskog programskog vodiča kao i dosta drugih novih pogodnosti koje pruža digitalna televizija. Razvoj digitalne televizije otvorio je mnoge mogućnosti za razvoj televizijskog prijemnika, usled čega dolazi do sve većeg korišćenja u druge svrhe pre svega zabave, igranje igrica, korišćenje interneta i društvenih mreža.

2.2 OTT sadržaj

U kontekstu slanja informacija OTT služi za slanje audio,video i ostalih sadržaja putem internet mreže bez potrebe uključivanja operatera sa mrežnom infrastrukturom kao posrednika u slanju. Za dobavljanje OTT sadržaja je potrebno samo imati pristup internet mreži. Zbog jednostavnosti pristupu OTT sadržaju ovaj vid komunikacije digitalnog STB (engl. Set Top Box) uređaja sa mrežnim poslužiocima je sve popularniji i iz godine u godinu sve se više koristi.

Sadržaj koji se nalazi na mrežnim poslužiocima određenog operatera može biti raznovrstan:

- Live IP – televizijski sadržaj koji se emituje uživo sa svim servisima kao i na digitalnim servisima (EPG, teletekst, zvučne trake)
- VOD – video ili audio sadržaj koji se pušta na zahtev korisnika, može biti besplatan ili da se plaća njegovo izdavanje
- Catch Up – sadržaj koji se emitovao na Live IP servisu do par dana (zavisi od mogućnosti poslužioca sadržaja) u prošlosti
- Netvork PVR – mogućnost trajnog snimanja Live IP sadržaja na mrežnom poslužiocu
- Aplikacije od "treće" strane - sadržaj se ne nalazi direktno na poslužiocu operatera već na poslužiocima samih aplikacija. Najpoznatiji takvi servisi su Hulu, HBO, Netflix, YouTube ...

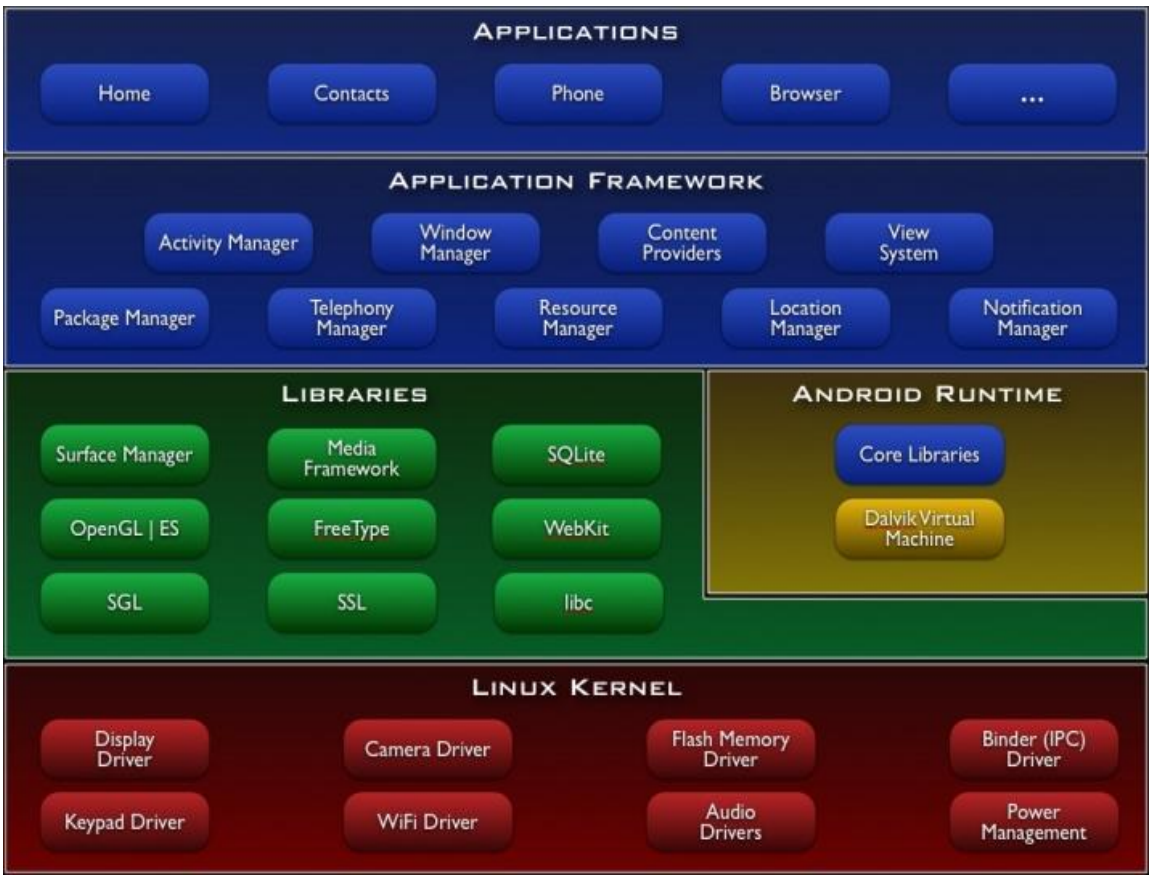
Najveća mana ovog sistema je ta da kvalitet servisa ne može da se garantuje pošto zavisi od internet mrežne infrastrukture korisnika digitalnog STB uređaja. Svaka smetnja na mreži može dovesti do smanjenja kvaliteta sadržaja koji se doprema sa poslužioca sadržaja.

2.3 Android platforma

Android operativni sistem je razvila grupacija firmi okupljena oko *Open Handset Alliance*, kao operativni sistem namenjen prevashodno mobilnim telefonima. Baziran je na Linuks operativnom sistemu pa se uspešno koristi i u drugim uređajima kao što su tablet, laptop i netbook računari, čitači elektronskih knjiga, digitalni televizijski prijemnici...

Na slici 2. prikazana je arhitektura Android operativnog sistema [4]:

- Linuks kernel nalazi se na dnu Android arhitekture i deluje kao sloj apstrakcije izmedju hardvera i ostatka steka. Kernel obezbeđuje programsku podršku za različite delove mobilnih uređaja kao i kontrolu paljenja. Upravljanje memorijom i procesima kao i mrežnim stekom je takođe u nadležnosti linuks kernela
- Biblioteke obezbeđuju skup C i C++ biblioteka koje se koristi od strane različitih komponenti Android operativnog sistema. MediaFramework biblioteka se koristi za skladištenje i reprodukciju multimedijalnih sadržaja. Sqlite je baza podataka dostupna svim aplikacijama. SGL biblioteka se u osnovi koristi za prikazivanje dvodimenzionalne grafike, dok se OpenGL ES koristi za trodimenzionalnu. Biblioteka Libc biće detaljno opisana u sledećem poglavlju.
- Android Runtime sadrži core biblioteke, koje obezbeđuju osnovnu funkcionalnost dostupnu u osnovnim bibliotekama Java programskog jezika, i Dalvik Virtual Machine čija je uloga da smanji trag memorije i da efikasno omogući rad više virtualnih mašina.



Slika 2: Arhitektura Android operativnog sistema

- Application Framework omogućava ponovnu upotrebu komponenti. Svaka aplikacija može da koristi mogućnosti drugih komponenti kao i da omogući korišćenje svojih. Svaka aplikacija u osnovi ima sledeće komponente: View System (sadrži dugmiće, liste, polja za unos teksta), Activity Manager (kontroliše kretanje i upravlja životnim ciklusom aplikacije), Resource Manager (omogućava pristup sadržajima kao što su grafike, izgledi, lokalni stringovi...), Content Provider (dozvoljava aplikaciji da podeli svoje podatke i da pristupa podacima drugih aplikacija).
- Application programi koji se dobijaju uz svaki Android uređaj. Kalendar, pretraživač, program za slanje tekstualnih poruka, proveru internet pošte.

3. Koncept rešenja

Cilj implementacije ovog rada je napraviti programsku podršku za reprodukciju digitalnog televizijskog i internet sadržaja u okviru Android operativnog sistema. Rešenje integracije zahteva postojanje više slojeva programske podrške prikazane na slici 3.

Slika 3: Struktura programske podrške DTT prijemnika zasnovanom na Android platformi

Glavni akcenat u ovom rešenju je stavljen na tzv. gornji deo programske podrske, na slici 3. obeležen narandžastom bojom, pisan Java programskim jezikom dok je donji deo podrške sa stanovišta ovog rešenja tzv. crna kutija (engl. *black box*), na slici 3. obeležena zelenom bojom.

3.1 MAL sloj

MAL je najniži deo programske podrške i predstavlja apstrakciju srednjeg sloja programske podrške i funkcija Android operativnog sistema. Ovaj sloj apstrakuje funkcionalnost programske podrške za televizijske prijemnike, fizičke arhitekture kao i funkcije operativnog sistema i predstavlja standardizovanu programsku spregu koja u komunikaciji sa programskom podrškom za televizijske prijemnike obezbeđuje DTT funkcionalnost višim slojevima.

3.2 Sloj usluga

Sloj usluga programske podrške sastoji se iz 4 posebna dela:

- LiveCentralService
- ServerCentralService
- ChannelListProvider
- EPGProvider

LiveCentralService servis predstavlja standardni Android servis. Služi kao sprega između grafičke korisničke sprege (eng. *GUI*) i MAL-a. Zadužen je za pripremu i prikupljanje podataka o DTT relevantnim stvarima kao sto su sadržaji elekronskog programskog vodiča (engl. Electronic Program Guide), pronađeni servisi, vreme iz servisa... Razlog njegovog uvođenja je prilagođavanje DTT aplikativne sprege zahtevima GUI aplikacije, kao i odvajanje logičkog rešenja od grafičkog.

ServerCentralService je još jedan Android servis koji se koristi kao sprega između GUI-a i posluživača OTT sadržaja. Zadužen je da na zahtev grafičke korisničke sprege pribavi sadržaje koji se nalazi na OTT poslužiocu: Live IP servise, EPG i Catch Up.

ChannelListProvider predstavlja dobavljača servisa i upravlja pristupom struktuiranom skupu podataka. Prihvata podatke od srednjeg sloja i sa OTT poslužioca i pruža mehanizam za definisanje sigurnosti podataka.

EPGProvider služi za prikupljanje DTT EPG podataka iz srednjeg sloja kao i EPG podataka sa OTT poslužioca za Live IP servise. Dobavljači sadržaja su standardna sprega, u Android operativnom sistemu, koja povezuje podatke iz jednog procesa sa drugim procesima. Dobavljač sadržaja prima zahteve sa podacima od klijenata, izvršava željenu akciju, i vraća rezultat.

3.3 GUI aplikacija

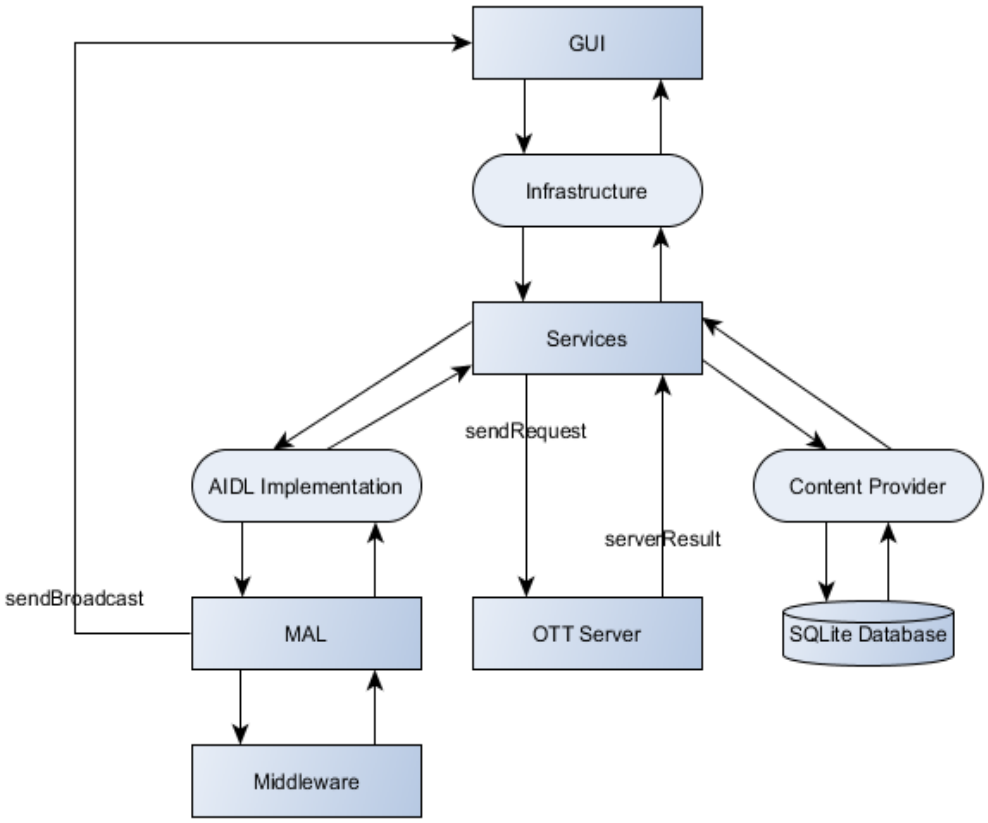
Grafička korisnička sprega je zamišljena kao TV aplikacija u kojoj korisnik korišćenjem daljinskog upravljača vrši interakciju sa DTT prijemnikom. Ona pokriva osnovne

funkcionalnosti koje DTT prijemnik treba da podržava kao što su: pretraživanje servisa, promena servisa, prikazivanje liste dostupnih servisa kao i prikazivanje informacija iz elektronskog programskog vodiča. Sastoji se od dve Android aktivnosti ScanActivity i LiveActivity.

3.4 Komunikacija između slojeva programske podrške

Jedan od najvećih problema koji se javlja pri izradi kompleksnog sistema programske podrške je povezivanje raznih delova sistema pa je usled toga potrebno definisati spregu kojom se odvija ova komunikacija. Na slici 4. je prikazana komunikacija između različitih slojeva programske podrške.

Kao sprega između grafičke korisničke sprege i sloja usluga napravljen je sloj infrastrukture. Deljiv je i sadrži klase koje koriste i grafička korisnička sprega i sloj usluga. Jedan od glavnih razloga njegovog uvođenja i korišćenja je da bi se izbeglo bespotrebno dupliranje koda u GUI i sloju usluga. Ovaj sloj je važan i za moguće proširenje ovog rešenja uvođenjem npr. aplikacije za reprodukciju VOD sadržaja sa mrežnog poslužioca koja bi koristila sloj infrastrukture za komunikaciju sa donjim delom sistema preko sloja usluga u kojem se nalazi između ostalog i komunikacija sa mrežnim poslužiocem.



Slika 4: Komunikacija između različitih slojeva programske podrške

AIDL (engl. Android Interface Definition Language) je poseban programski jezik kojim se definiše programska sprega koja enkoduje i dekoduje podatke. Pomoću njega se povezuje sloj usluga sa najnižim delom programske podrške (MAL) preko Binder mehanizma komunikacije.

Sloj usluga sa bazom podataka je povezan dobavljačem sadržaja koji omogućava standardnu spregu za pristup podacima u Android operativnom sistemu. Dobavljač sadržaja prima zahteve za podacima od sloja usluga, izvršava željenu akciju, i vraća rezultat.

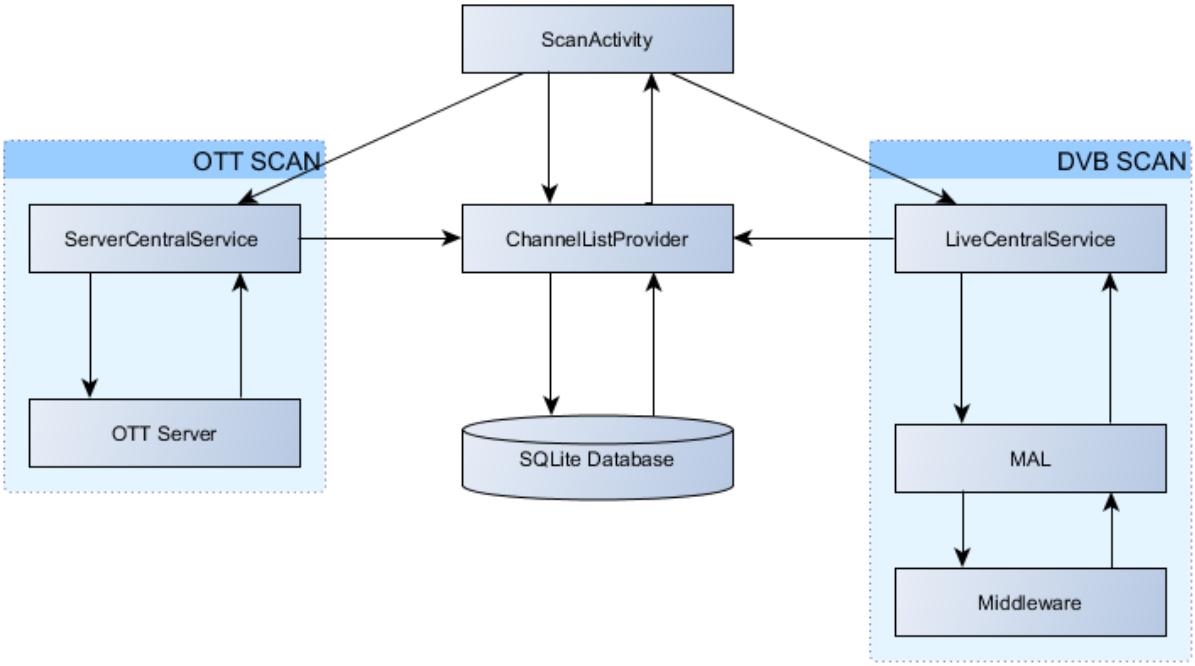
Za komunikaciju i razmenu podataka sloja usluga sa mrežni poslužiocem koristi se JSON (engl. JavaScript Object Notation) standard. On je vrlo jednostavan za korišćenje i nezavisan je od programskog jezika.

3.5 Pretraživanje DTT i OTT servisa

Pretraga servisa predstavlja prikupljanje liste servisa prelaskom kroz ceo frekventni opseg i pronalaženjem svih postojećih servisa u slučaju DTT pretraživanja. Pretraga celokupnog skupa servisa sastoji se od poziva instrukcije za pretragu servisa i upisa pronađenih servisa u bazu podataka. Procedura pretrage DTT servisa je dosta komplikovanija od ovog prikazanog rešenja, ali celokupnu proceduru nije moguće prikazati zato što je realizacija sakrivena u srednjem sloju programske podrške koja je dobijena kao gotovo rešenje.

OTT pretraživanje je dosta jednostavnije i sastoji se od slanja zahteva mrežnom poslužiocu, prihvatanju i obradi odgovora i u slučaju merodovnog odgovara vrši se upis dobijenih servisa u bazu podataka.

Na slici 5. je prikazana realizacija pretraživanja servisa.



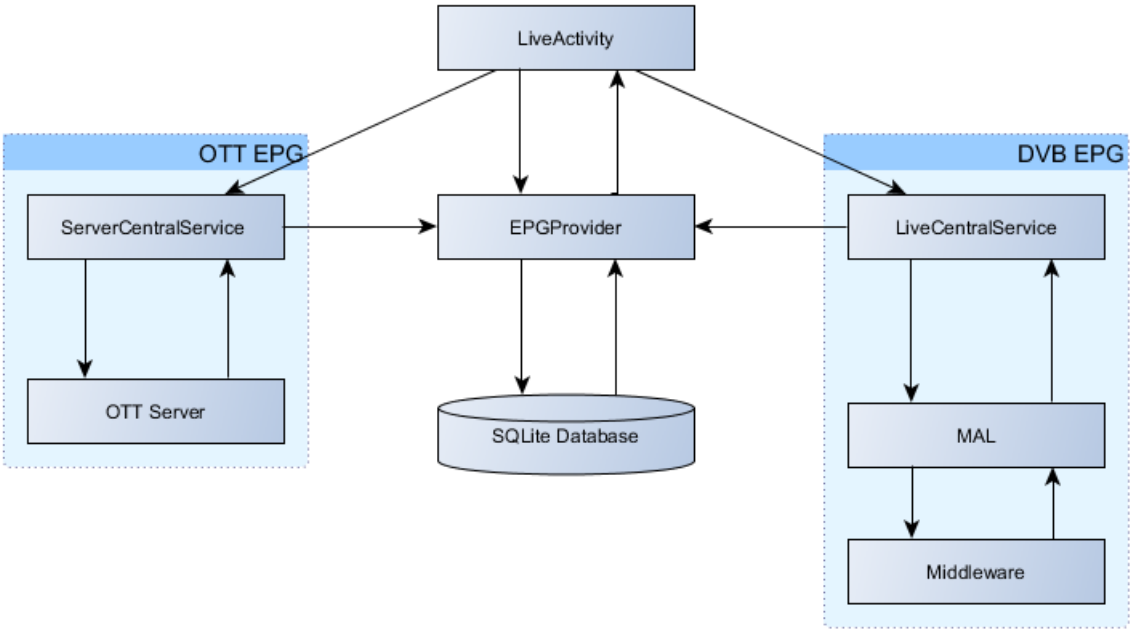
Slika 5: Prikaz pretraživanja servisa

3.6 Prikupljanje elektronskog programskog vodiča

Posle uspešno izvršenog prikupljanja servisa potrebno je pribaviti elektronski programski vodič za pronađene servise. Kao i u slučaju pretraživanja servisa i mehanizam prikupljanja DTT elektronskog programskog vodiča je dobijen kao već implementirana programska podrška. Pretraga elektronskog programskog vodiča DTT servisa se ne može vršiti na celom frekventnom opsegu tj. za sve pronađene servise odjednom već se vrši samo na frekvenciji DTT servisa koji se trenutno prikazuje. Sa strane aplikacije dovoljno je poslati frekvenciju servisa koji se trenutno prikazuje i vremenski opseg za koji se zahteva elektronski programski vodič. Po dobijanju rezultata sadržaj se smešta u bazu podataka.

Prikupljanje elektronskog programskog vodiča za OTT servise vrši se slanjem JSON zahteva mrežnom poslužiocu. Zahtev treba da sadrži jedinstveni broj servisa, koji se dobija posle završetka prikupljanja servisa, i vremenski opseg za koji se zahteva elektronski programski vodič. Za ove servise je moguće dobiti i elektronski programski vodič za emisije iz prošlosti i gledati Catch Up sadržaje koji su u bili prikazani, ali se u ovom radu taj slučaj neće razmatrati.

Na slici 6. je prikazana realizacija prikupljanja elektronskog programskog vodiča.



Slika 6: Realizacija prikupljanja elektronskog programskog vodiča

4. Programsko rešenje

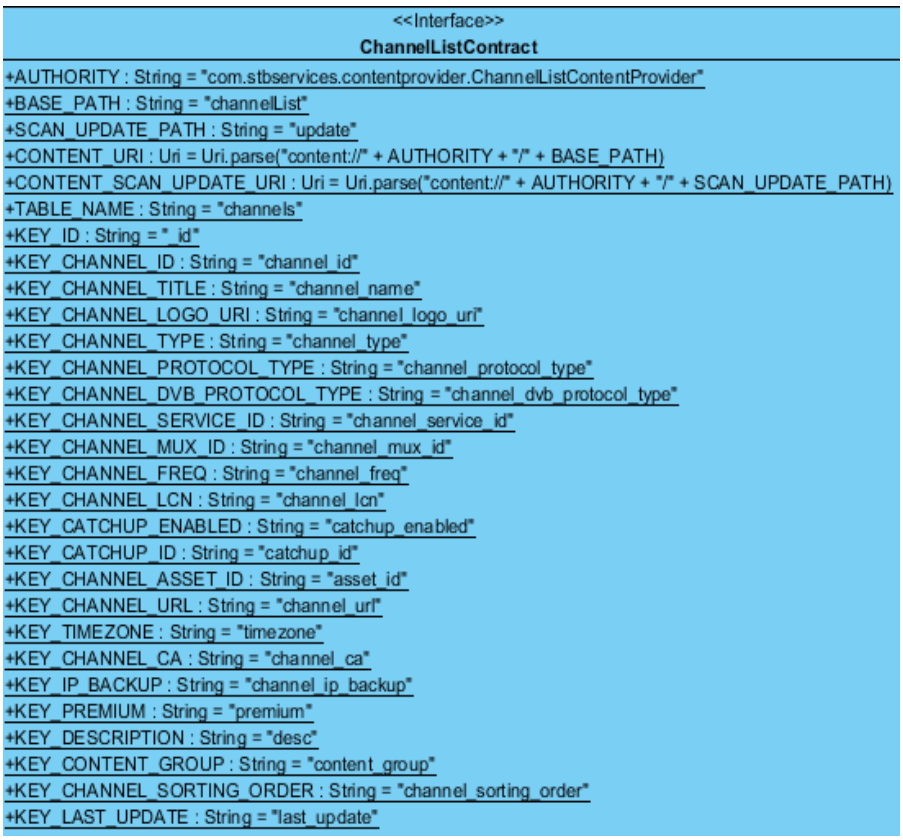
U ovom poglavlju biće opisana realizacija slojeva programske podrške, njihova međusobna komunikacija, realizacija pretraživanja servisa kao i prikupljanja elekronskog programskog vodiča.

4.1 Sloj infrastrukture

Služi kao sprega između grafičke korisničke sprege i sloja usluga. Sastoji se iz dva izdvojena dela: snabdevač modul koji služi za komunikaciju sa delom sloja usluga vezanim za bazu podataka i modul usluga koji služi za komunikaciju sa ostalim delovima sloja usluga.

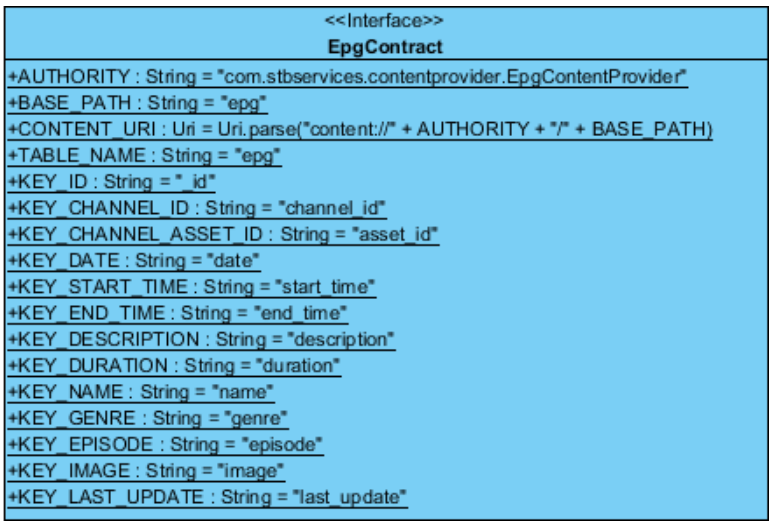
4.1.1 Snabdevač modul

Sadrži dve java programske sprege, jedan za listu servisa a drugi za elektronski programski vodič. ChannelListContract sadrži sva polja koja su potrebna, grafičkoj korisničkoj sprezi, da bi se dobavili servisi iz baze podataka odnosno prikupile informacije o njima kao što su ime servisa, tip, frekvencija za DTT servise odnosno jedinstveni broj servisa, URL adresa servisa i ostalo za OTT servise. Na slici 7. prikazan je UML dijagram ChannelListContent programske sprege.



Slika 7: UML prikaz ChannelListContract programske sprege

Za prenos podataka i dobavljenje sadržaja iz baze elektronskog programskog vodiča kao sprega između GUI i Service sloja koristi se EpgContract programska sprega. Ona sadrži polja koja su neophodna grafičkoj korisničkoj sprezi za prikaz elektronskog programskog vodiča kao što su ime emisije, dužina, opis... Na slici 8. je prikazan UML dijagram EpgContract programske sprege.



Slika 8: UML prikaz EpgContract programske sprege

4.1.2 Modul usluga

Organizovan je u tri različita dela. Prvi služi za komunikaciju sa delom sloja usluga koji komunicira sa MAL slojem i sastoji se od više AIDL fajlova. Neke od funkcija koje prolaze kroz ovaj sloj su:

- **boolean** playChannel(**long** channelId);
- **void** startAutoScan(in **int**[] freqTable);
- **void** stopScan();
- **int** getSignalQuality();
- **int** getSignalStrength();
- **int** getScanStatus();
- **long** getCurrentChannelId();
- **List** getEpgByChannelId(**long** channelId, **long** startDate, **long** endDate);

Ono što se može zaključiti iz spiska funkcija koje se pozivaju da se kroz ovaj sloj ostvaruje osnovna funkcionalnost digitalnog televizijskog prijemnika bez naprednih funkcionalnosti i internet mrežnog sadržaja. Uz pomoć ovog sloja, sa stanovišta grafiške korisničke sprege, veoma se lako mogu implementirati sve te funkcionalnosti.

Sledeći deo u modulu usluga sloja infrastrukture služi kao komunikaciona sprega sa delom sloja usluga koji komunicira sa mrežnim poslužiocem. Ovaj deo je kompleksan i sastoji se od više različitih klasa, AIDL fajlova i nabrajanja. U daljem tekstu biće objašnjeni samo glavni delovi bez kojih komunikacija grafičke korisničke sprege sa mrežnim poslužiocem nije moguća. ServerCentral.aidl se sastoji od sledećih funkcija koje se implementiraju u sloju usluga:

- **void** resolveServer(**String** name, in **ServerCallback** callback);
- **void** sendRequest(in **JsonObject** request, in **ServerCallback** callback);
- **int** addEventCallback(**int** eventCode, in **ServerEventCallback** callback);
- **void** removeEventCallback(**int** callbackId);

Povezivanje grafičke korisničke sprege sa mrežnim poslužiocem počinje pozivom resolveServer funkcije koja služi za povezivanje sa mrežnim poslužiocem pri čemu se vrši registracija svih poziva ka njemu. Posle uspešno uspostavljene veze sa mrežnim poslužiocem moguće je slati zahteve ka istom pozivanjem sendRequest funkcije. Pri slanju zahteva potrebno je registrovati callback koji će se, po završetku obrade zahteva na mrežnom poslužiocu, asinhrono pozvati i doneti rezultat. Lista svih podržanih zahteva koji se mogu poslati mrežnom poslužiocu nalazi se u nabrajanju ServerAction. Najvažniji zahtevi koji su se koristili pri izradi ovog rešenja su zahtev za dobijanje liste servisa i zahtev za dobijanje elektronskog programskog

vodiča. U slučaju neuspešnog pokušaja dobavljanja podataka sa mrežnog poslužioca potrebno je pravilno reagovati na grešku. Spisak kodova greški, koje se mogu dogoditi, nalazi se u nabrajanju ServerError.

Treći deo modula usluga upotpunjuje komunikaciju između grafičke korisničke sprege i sloja usluga. Tu se nalaze apstraktne funkcije koje se odnose i na vezu sa mrežnim poslužiocem i na vezu sa srednjim slojem kao i kombinaciju ta dva. Deo funkcija koje prolaze kroz ovaj sloj su:

- **void** getSecuredUri(**LiveCentralSecureIPListener** listener, **String** asset_id);
- **boolean** getCatchupUri(**LiveCentralCatchupListener** listener, **long** time, **String** asset_id);
- **boolean** refreshChannelList();
- **boolean** refreshSIEpgData();

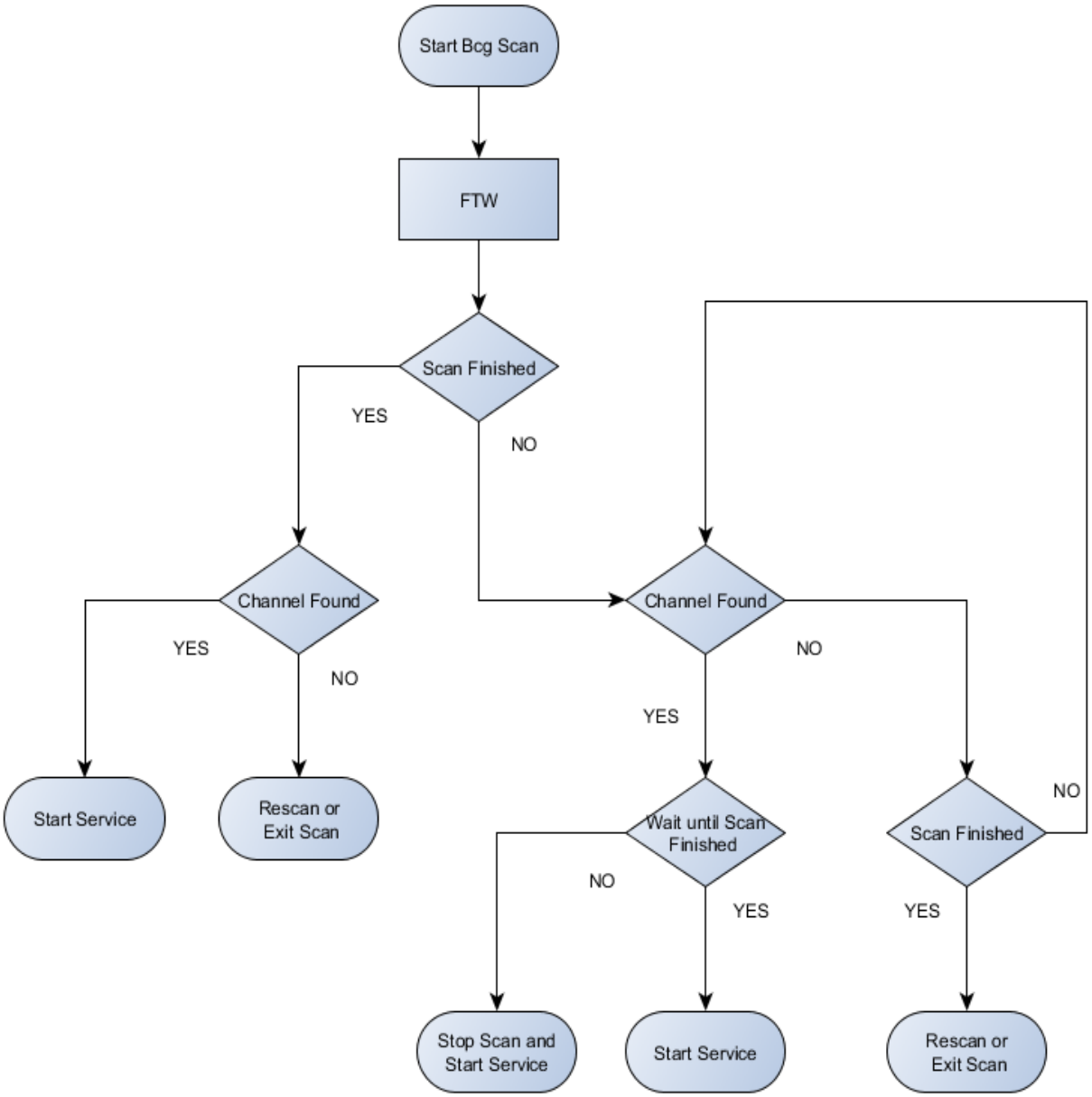
4.2 Pretraga servisa

Za pretragu servisa implementirana je posebna komponenta korisničke grafičke sprege ScanActivity. Realizovana komponenta grafičke korisničke sprege za pretragu servisa pokreće se pri potrebi za pretragom servisa iz podešavanja aplikacije odnosno pri prvom korišćenju uređaja kao pozadinski proces dok traje početno podešavanje uređaja (engl. First Time Wizard).

Pretraga servisa digitalne televizije je vremenski zahtevna operacija pa se oslanja na mehanizam asinhrone programske podrške. Uz pomoć mehanizma asinhrone programske sprege grafička korisničke sprege za pretragu servisa dobija informacije o pronađenim servisima kao i o završetku procesa pronalaženja digitalnih servisa i pristupa pronalazenju servisa koji se nalaze na mrežnom poslužiocu.

4.2.1 Pretraga servisa pri prvom korišćenju uređaja

Da bi se za krajnjeg korisnika uređaja ubrzao proces pronalaženja servisa, napravljena je pretraga servisa koja se izvršava u pozadini dok korisnik podešava početne postavke uređaja. Na ovaj način je omogućeno da se po završetku početnih podešavanja uređaj odmah koristi bez dodatnih radnji i dodatnog čekanja. Na slici 9. je prikazan blok dijagram algoritma za pretragu servisa u pozadini.



Slika 9: Blok dijagram algoritma za pretragu servisa u pozadini

U sloju grafičke korisničke sprege nalazi se jedan android servis (ScanService) koji rukovodi pretragom servisa. Ovaj servis se pokreće na paljenju uređaja. Pri prvom pokretanju uređaja ScanService se pokrene i započinje pretragu servisa u pozadini. Pretraga servisa digitalne televizije započinje pozivom funkcije:

```
public void startScan(LiveScanListener listener, int frequency)
```

Ulazni parametri:

1. LiveScanListener listener – osluškivač događaja tipa LiveScanListener
2. int frequency – frekvencija servisa koji se pretražuje u slučaju ručnog biranja servisa(ovaj slučaj se ne razmatra u ovom rešenju) dok se u slučaju pretrage celog

opsega frekvencija salje vrednost nula (0) što predstavlja indikaciju da je potrebna pretraga celog opsega

Povratna vrednost: nema

LiveScanListener predstavlja najveću apstrakciju funkcija pretrage sa strane grafičke korisničke sprege. Da bi se grafički prikaz pretrage realizovao potrebno je samo pozvati funkciju za početak pretrage i osluškivati ovaj listener na pravi način. On obaveštava o sledećim događajima:

1. promeni jačine signala: **public void** onSignalStrengthUpdate(**int** signalStrength)
2. promeni kvaliteta signala: **public void** onSignalQualityUpdate(**int** signalQuality)
3. progresu pretrage servisa: **public void** onScanProgressUpdate(**int** progress, **int** progressMax)
4. pronađenim servisima: **public void** onChannelFound(ScannedChannelItem channel) pri čemu se u ScannedChannelItem nalaze podaci o imenu, tipu i LCN-u (engl. Logical Channel Number) servisa

Sa algoritma sa slike 9. moze se zaključiti da postoji više mogućnosti za nastavak i završetak pretrage. Po završetku početnog podešavanja uređaja u slučaju da se pretraga servisa završila i ako je pronađen bar jedan servis pokreće se reprodukcija prvog pronađenog servisa. U slučaju da se pretraga servisa završila i da ni jedan servis nije pronađen korisniku se daje mogućnost da ponovno pokrene pretraživanje servisa ili da koristi ostale funkcije uređaja bez digitalnog televizijskog sadržaja.

Drugi slučaj je dosta kompleksniji i on se dešava kad posle završetka početnog podešavanja uređaja pretraga servisa nije završena. Pri pokretanje pretrage u pozadini u ScanService-u se uvodi bcgScanListener, kao objekat LiveScanListener-a, koji prima informacije o pretrazi servisa. Kada se završi početno podešavanje uređaja pokreće se grafička korisnička sprega dijaloga pretrage ScanFragment. Informacije o statusu pretrage, pronađenim servisima (ako ih ima) i progresu pretrage zahteva pozivom sledećih funkcija iz ScanService-a:

public int scanStatus()

Ulazni parametri: nema

Povratna vrednost: tipa int i može biti jedna od sledeće tri vrednosti:

1. SCAN_FINISHED = 0 – pretraga servisa je završena
2. SCAN_IN_PROGRESS = 1 – pretraga servisa je u toku
3. BCG_SCAN_FINISHED = 2 – pretraga servisa u pozadini je završena

public int getProgress()

Ulazni parametri: nema

Povratna vrednost: tipa int i vrednost je u rasponu od 0 do 100

public List<ScannedChannelItem> getFoundChannels()

Ulazni parametri: nema

Povratna vrednost: lista ScannedChannelItem tipa koja sadrži tri polja:

1. LCN – LCN servisa
2. Name – ime servisa
3. Type – tip servisa (TV ili RADIO)

Po završetku prikupljanja informacija o pretrazi, ScanFragment prijavljuje novi LiveScanListener koji nastavlja da prikuplja nove podatke o pretrazi servisa. Od ovog trenutka pretraga servisa se vrši na isti način kao i pretraga servisa iz podešavanja aplikacije, pa će taj deo biti opisan u daljem tekstu.

4.2.2 Pretraga servisa iz menija podešavanja aplikacije

Drugi način pretrage servisa je pokretanjem iz menija podešavanja aplikacije. Pretraga digitalnih zemaljskih servisa se može vršiti samo za ceo frekvetni opseg, pri čemu su podržana dva standarda digitalne televizije DVB-T i DVB-T2.

Pretraga se pokreće, kao što je bilo opisano, pozivom funkcije startScan(**LiveScanListener** listener, **int** frequency) iz ScanFragmenta. U njoj se registruje prijemnik (engl. receiver) tipa LiveScanReceiver(Context context, LiveScanListener listener). Ovaj prijemnik proširuje androidovu klasu BroadcastReceiver, čija je uloga da prihvati sve događaje namere poslate funkcijom sendBrodcast(Intent intent). U LiveScanReceiver-u se vrši selekcija događaja tako što postavlja se filter koji ograničava primanje samo događaja vezanih za pretragu servisa, a to su MSG_SCAN_INFO i MSG_PROG_INFO.

MSG_SCAN_INFO i MSG_PROG_INFO nose informacije kojima se podešava listener, na koji se reaguje u grafičkoj korisničkoj sprezi. MSG_SCAN_INFO sadrži informacije o jačini i kvalitetu signala kao i o progresu pretrage servisa dok se u MSG_PROG_INFO nalazi ime servisa, LCN i tip.

Poslednji sloj programske podrške MAL je najsloženiji. Komunicira sa middleware-om i prikuplja sve informacije vezane za proces pretrage i njih prenosi višim slojevima slanjem poruka mehanizmom namere tj. pozivom funkcije sendBrodcast(Intent intent). U TerControl

klasi se nalazi logika prikupljanja informacija o pretrazi servisa. Na inicijalizaciji se registruje dobavljač poruka iz middleware-a. Događaji na koje se reaguju su sledeći:

1. EVENT_SEARCH_END – završetak procesa prikupljanja servisa na zaključanoj frekvenciji
2. EVENT_HW_TUNER_LOCK – tjuner je zaključan na frekvenciju
3. EVENT_HW_TUNER_UNLOCK – tjuner je otključan sa frekvencije

Kada se tjuner zaključa na frekvenciju, prvo se proveri da li je kvalitet signala dovoljno dobar za uspešnu pretragu frekvencije i u slučaju da jeste računa se progres pretrage na sledeći način:

```
if (getSNR().SNR>10) {
  if (G_freq.length > 1) {
    double freq_num=G_freq.length;
    Percent = (int) ((100 / freq_num) * Freq_index);
    if(Freq_index == G_freq.length){
      Percent = 100;
    }
  } else {
    Percent = 0;
  }
  SignalQuality = getSNR().SNR;
  SignalStrength = getSignalStrength();

  mDB.sendFrontendInfo(Percent, SignalQuality, SignalStrength);
```

Posle izračunavanja progresa od srednjeg sloja se zahtevaju informacije o kvalitetu i jačini signala. Kada se prikupe svi podaci oni se šalju privremenoj lokalnoj bazi koja se koristi dok je prikupljanje podataka u toku. U funkciji sendFrontendInfo(int percent, int signalQuality, int signalStrength) se vrši prepakivanje dobijenih podataka u String i njihovo slanje mehanizmom namere kao događaj MSG_SCAN_INFO.

Po završetku skupljanja servisa na datoj frekvenciji reaguje se na događaj EVENT_SEARCH_END pri čemu se pravi lista sa svim pronađenim servisima i čuvaju se informacije o njima u lokalnoj bazi. Po završetku obrade šalju se događaji MSG_PROG_INFO koji sadrže informacije o nazivima, tipu i LCN-u pronađenih servisa na sledeći način:

```
for (int i = 0; i < ServiceCount; i++) {

 ProgData data = getServiceData(i);

  if (data != null) {
     mDB.sendDTVType(data.ServiceType, data.ProgramName, data.LogicNum);
  }
}
```

Kada progres pretrage dostigne maksimalnu vrednost, odnosno kada se pretraga završila kreće se u proces dobavljanja svih pronađenih servisa iz privremene baze koja se nalazi u MAL sloju i punjenja glavne baze podataka. Po završetku punjenja baze, u slučaju da je uređaj povezan na mrežu šalje se zahtev mrežnom poslužiocu za dobijanje liste dostupnih OTT servisa:

```
public static void performServerChannelListGet() throws RemoteException {
    JsonObject contentGroupRequest = ServerActionBuilder.liveChannels();

    mServer.sendRequest(contentGroupRequest, new ServerCallback.Stub() {
        @Override
         public void onServerResult(ServerResult result) throws RemoteException {
                if (result.isSuccess()) {
                    JsonArray jsonArray = result.getResultAsArray();
                    DBUpdateManager.getInstance(mContext).updateChannelList(jsonArray,
null);
                }

        mContext.sendBroadcast(new
Intent(LiveDefines.getEventBroadcastAction(LiveEvent.UpdateChannelList)));
        }
    });
}
```

Po dobijanju rezultata od mrežnog poslužioca ažurira se lista servisa dodavanjem OTT servisa. Poslednji korak kojim se završava proces pretrage je generisanje liste servisa po sledećem pravilu:

1. DTT TV servisi
2. DTT TV servisi
3. DTT RADIO servisi
4. u slučaju servisa na mreži, servisi sa lošijom jačinom signala idu na kraj liste

4.3 Prikupljanje elektronskog programskog vodiča

Prikaz elektronskog programskog vodiča je osnovna funkcionalnost svakog uređaja digitalne televizije. U ovom rešenju biće prikazano dobavljanje za DTT i OTT servise.

4.3.1 Prikupljanje elektronskog programskog vodiča za OTT servise

Postoje dva slučaja slučaja kada se pribavlja elektronski programski vodič sa mrežnog poslužioca. Prvi slučaj je po završetku pretrage digitalnih servisa kada se pošalje sledeći zahtev za dobijanje sadržaja :

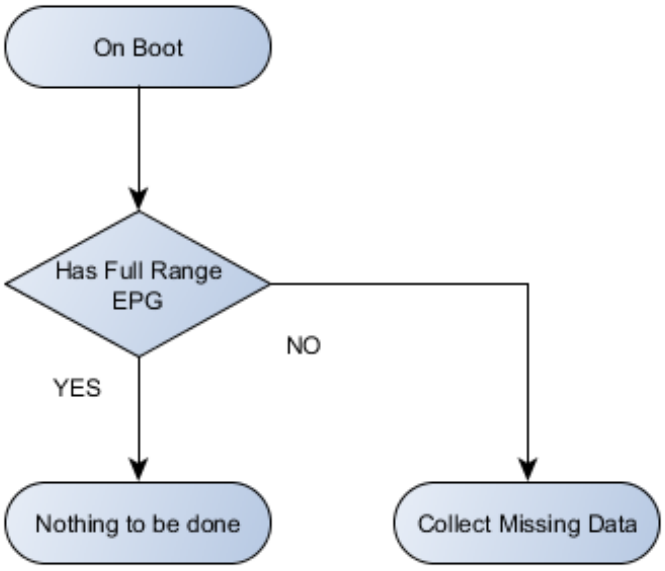
```
public static void performServerEPGEventsGet(String channelAssetId,
                                long epgEventsFrom, long epgEventsTo) throws RemoteException {

    JsonObject contentGroupRequest = ServerActionBuilder.liveChannelEpg(
          channelAssetId, epgEventsFrom, epgEventsTo);

    mServer.sendRequest(contentGroupRequest, new ServerCallback.Stub() {
       @Override
       public void onServerResult(ServerResult result) throws RemoteException {
          if (result.isSuccess()) {
             JsonArray jsonArray = result.getResultAsArray();
             String assetId = AssetEpgEventHelper.getAssetId(result.getRequest());
            String startTime = AssetEpgEventHelper.getRequestStartTime(result.getRequest());
            DBUpdateManager.getInstance(mContext).updateEpgEvents(assetId, startTime, jsonArray);
           }

           DBUpdateManager.getInstance(mContext).getEpgEventHandle().sendEmptyMessage(0);
    }
  });
}
```

Kao što se može videti iz koda, za dobavljanje elektronskog programskog vodiča je potrebno poslati identifikacioni broj servisa, vreme od kada zahtevamo i do kada zahtevamo elektronski programski vodič. Po dobijanju odgovora šalje se zahtev rukovaocu baze da doda sadržaj u bazu podataka. Postoje dva načina za osvežavanje baze elekronskog programskog vodiča sa novim sadržajem. Na svakom pokretanju uređaja proverava se da li je baza puna za vremenski period od 7 dana i ako nije traži se od mrežnog poslužioca sadržaj za period koji nedostaje. Ovaj proces je prikazan na slici 10.

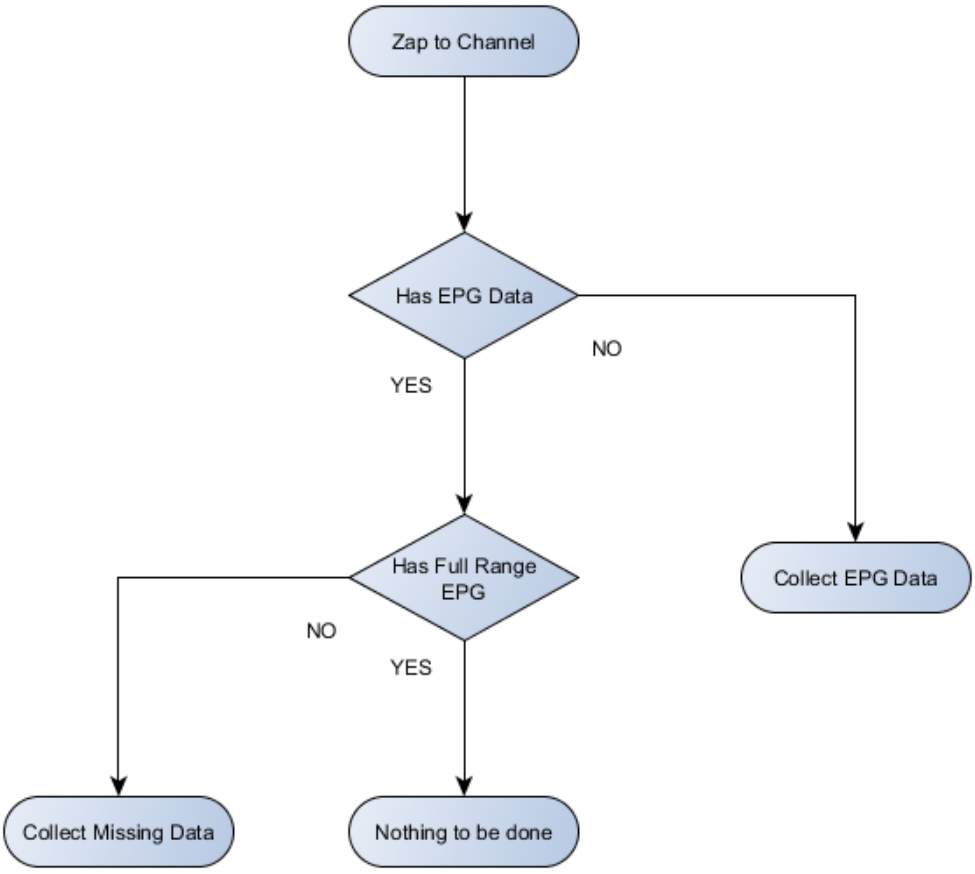


Slika 10: Blok dijagram algoritma za prikupljanje EPG-a za OTT servise na pokretanju uređaja

Ako uređaj radi više sati, potrebno je takođe napuniti bazu sa novim sadržajem pa se na svakih 12 sati zahteva od mrežnog poslužioca elektronski programski vodič za nedostajući vremenski period.

4.3.2 Prikupljanje elektronskog programskog vodiča za DTT servise

Za razliku od pretrage servisa, za DTT servise, koja prikupi sve servise u mreži, elektronski programski vodič je moguće dobiti samo za trenutni servis odnosno za sve servise na toj frekvenciji. Na slici 11. je prikazan blok dijagram algoritma za prikupljanje elekronskog programskog vodiča za DTT servise.



Slika 11: Blok dijagram algoritma za prikupljanje EPG-a za DTT servise

Kada se završi pokretanje servisa pristupa se proveri i pokretanju skupljanja elektronkog programskog vodiča ukoliko je potrebno. Sa strane grafičke korisničke sprege potrebno je pozvati funkciju **boolean** refreshSIEpgData() iz sloja infrastrukture. Ova funkcija je implementirana u sloju usluga. Iz srednjeg sloja se zahteva dobijanje identifikacionog broja servisa koji se trenutno reprodukuje. Na osnovu njega se iz baze servisa dobija frekvencija tog servisa. Dobijanje svih servisa sa date frekvencije se vrši upitom u bazu na sledeći način:

```
String selection = " channel_mux_id = " + mux_id + " AND asset_id = '"";

Cursor cursor = mContext.getContentResolver().query(ChannelListContract.CONTENT_URI,
null, selection, null, null);
if(cursor != null) {
  if (cursor.moveToFirst()) {
    do {
          int channel_id_index =
cursor.getColumnIndex(ChannelListContract.KEY_CHANNEL_ID);
          channel_id = cursor.getInt(channel_id_index);
          mEpgData = mMalCentral.getEpgByChannelId(channel_id, startTime, endTime);
          mContext.getContentResolver().bulkInsert( EpgContract.CONTENT_URI, mEpgData);

    } while (cursor.moveToNext());
   }
  cursor.close();
}
```

Android Cursor klasa se koristi za manipulaciju bazom podataka. Prvo se napravi upit Cursor-u da pokazuje u bazi samo na servise koji imaju istu frekvenciju. Ako upit u bazu prođe uspešno, Cursor postavljamo na prvi servis u listi i nad tim servisom zahtevamo prikupljanje elektronskog programskog vodiča pozivanjem funkcije iz MAL sloja programske podrške ArrayList<BaseEPGEntity> getEPGByChannelId(long channelId,long startDate, long EndDate). Po dobijanju odgovora za dati servis puni se baza sa sadržajem elektronskog programskog vodiča. Ovaj proces se ponavlja za sve servise na istoj frekvenciji.

5. Ispitivanje i rezultati

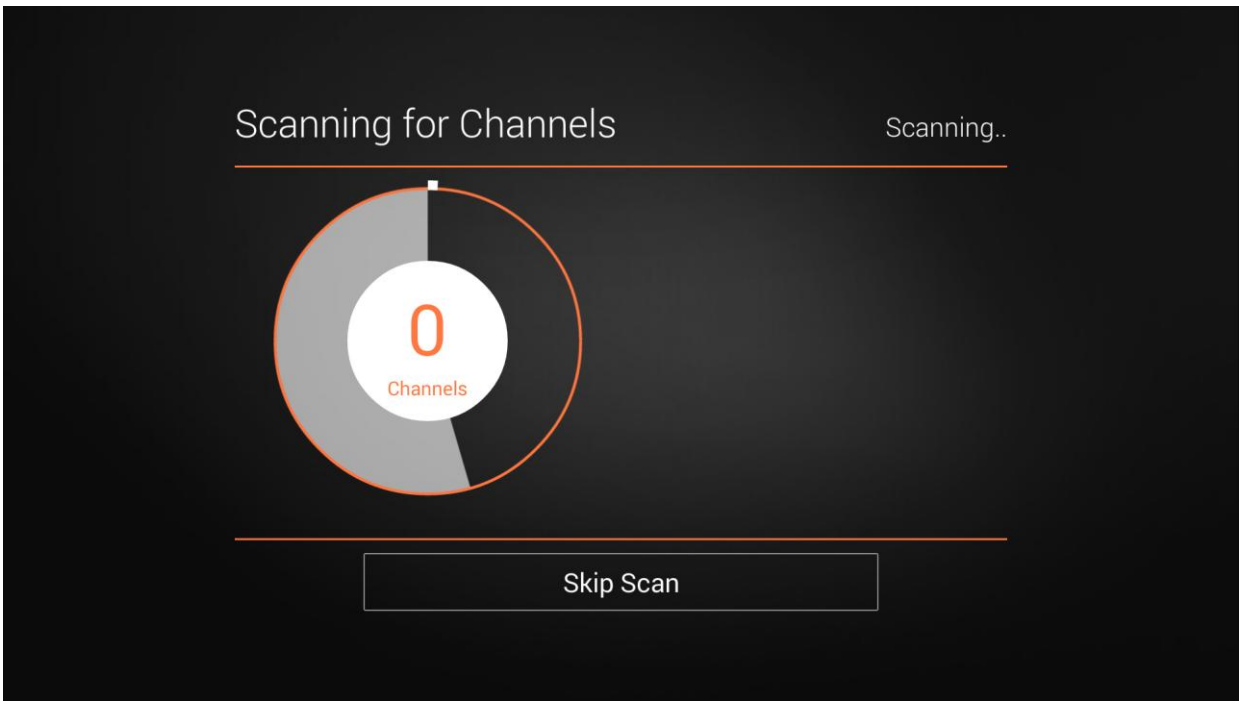
U ovom poglavlju dat je opis načina ispitivanja i verifikacije programskog rešenja, kao i rezultata koji su prikupljeni prilikom demonstracije funkcionalnosti.

5.1 Ispitivanje pretrage servisa

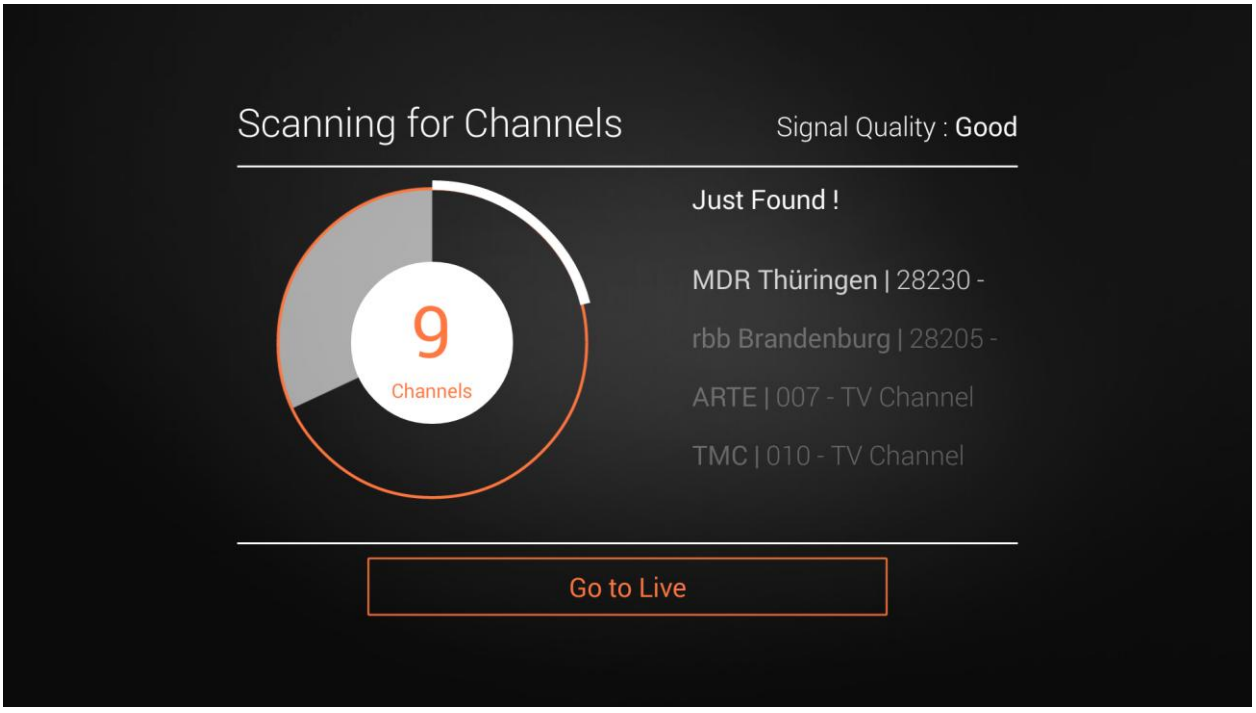
Kako bi se ispitalo programsko rešenje potrebno je pokrenuti proces pretrage i nakon toga prikazati dobijeni sadržaj. Proverom frekvencija, u slučaju pretrage DTT servisa, na kojima se nalazi poznat sadržaj moguće je utvrditi da li je pronalaženje servisa bilo uspešno, kao i prikaz sadržaja datog servisa na ekranu. Kod OTT pretrage servisa, provera dobijenih servisa se vrši upoređivanjem vrednosti koje su dobijene korišćenjem Postman mrežne aplikacije koja se koristi za proveru odgovora poslatih kao JSON zahtevi mrežnom poslužiocu.

Postoje tri stanja grafičke korisničke sprege u toku pretrage servisa koje treba verifikovati:

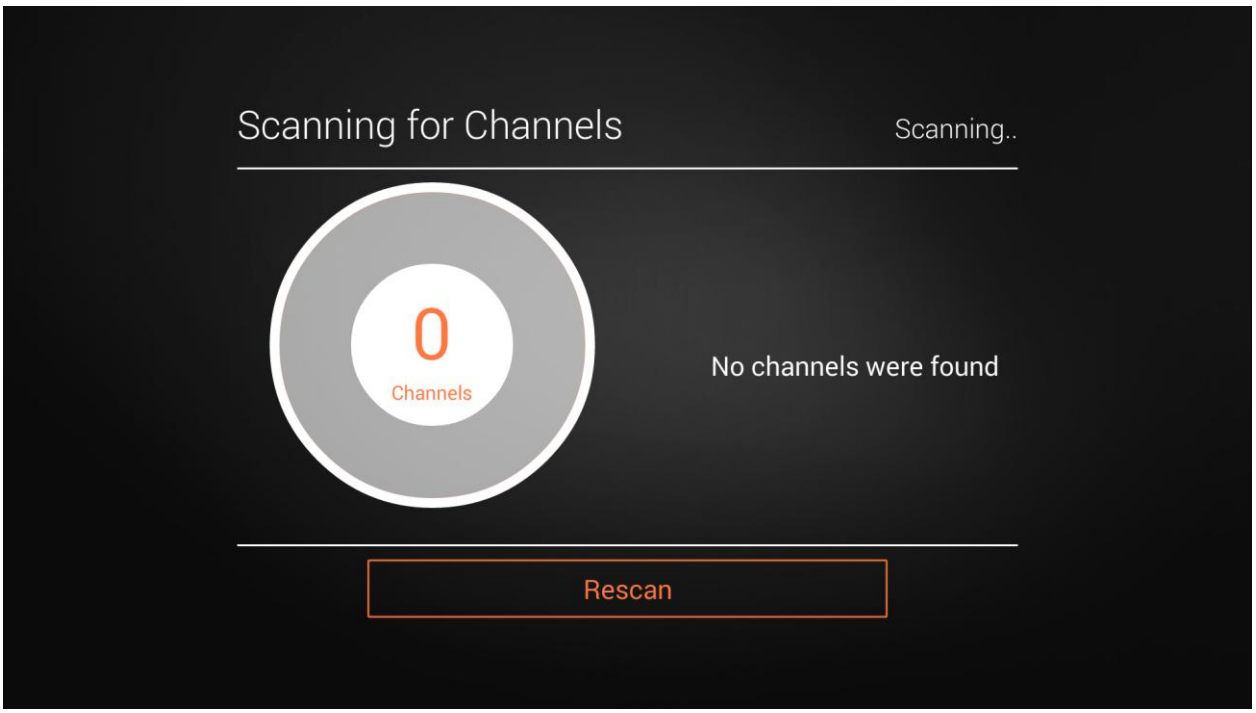
1. Pri pokretanju pretrage servisa i pre pronalaženja prvih servisa, korisniku se treba omogućiti da prekine proces pretrage. Izgled grafičke korisničke sprege procesa pretrage za ovaj slučaj prikazan je na slici 12.
2. Posle pronalaženja prvih servisa korisniku treba dati mogućnost da prekine proces pretrage, ako želi, i da započne reprodukciju servisa. Na slici 13. je prikaz ovaj slučaj.
3. U slučaju ne pronalaženja ni jednog servisa, korisniku treba dati mogućnost da ponovo pokrene proces pretrage servisa. Izgled grafičke korisničke sprege procesa pretrage za ovaj slučaj prikazan je na slici 14.



Slika 12: Početak pretrage servisa



Slika 13: Prikaz pronađenih servisa tokom pretrage



Slika 14: Prikaz završenog procesa pretrage bez pronađenih servisa

Verifikacija procesa pretrage je obavljena tako što su napravljena četiri testna slučaja. Pre početka procesa verifikacije, za potrebe testiranja, napravljena je konfiguracija lokalne DTT mreže sa dve frekvencije i ukupno dvadeset servisa na njima, dok se proverom mrežnog poslužioca Postman aplikacijom dobio rezultat od četiri OTT servisa.

Prvi testni slučaj: na uređaj koji se testira prikačen je antenski kabal i uređaj je povezan na internet mrežu. Po završetku procesa pretrage uređaj je pronašao ukupno 24 servisa i test se smatra uspešnim jer su svi servisi pronađeni.

Drugi testni slučaj: na uređaj koji se testira prikačen je samo antenski kabal. Po završetku procesa pretrage uređaj je pronašao ukupno 20 servisa i test se smatra uspešnim jer su svi DTT servisi pronađeni.

Treći testni slučaj: uređaj koji se testira povezan je samo na internet mrežu. Po završetku procesa pretrage uređaj je pronašao ukupno 4 servisa i test se smatra uspešnim jer su svi OTT servisi pronađeni.

Četvrti testni slučaj: na uređaj koji se testira nije prikačen ni antenski kabal niti je uređaj povezan na mrežu. Po završetku procesa pretrage uređaj nije pronašao ni jedan servis i test se smatra uspešnim jer je očekivano da se ne pronađe ni jedan servis.

Po završetku izvršenja ovih opisanih testnih slučajeva, izvršena je promena konfiguracije lokalne mreže odnosno promenjen je signal koji se šalje u mrežu na sledeći način:

1. Prva frekvencija DVB-T signal, druga DVB-T signal
2. Prva frekvencija DVB-T signal, druga DVB-T2 signal
3. Prva frekvencija DVB-T2 signal, druga DVB-T signal
4. Prva frekvencija DVB-T2 signal, druga DVB-T2 signal

Promenom konfiguracije lokalne DTT mreže broj testnih slučajeva koji su ispitani je povećan na šesnaest. Svi testovi su se uspešno izvršili na već opisan način.

Uređaj poseduje dva načina povezivanja na internet mrežu preko žice i bežično. Usled te činjenice testiranje je prošireno pa je ukupan broj testnih slučajeva za proces pretrage servisa trideset i dva.

5.2 Ispitivanje pribavljanja elektronskog programskog vodiča

Kako bi se ispitalo programsko rešenje za dobijanje elektronskog programskog vodiča potrebno je uspešno završiti proces pretrage servisa i početi reprodukciju DTT servisa. Dobijeni elektronski programski vodič se mora uporediti sa komercijalnim uređajem i na taj način se može verifikovati da li je dobar sadržaj dobavljen. U slučaju OTT servisa provera dobavljenog elektronskog programskog vodiča se vrši upoređivanjem vrednosti koje su dobijene korišćenjem Postman mrežne aplikacije. Na slici 15. se može videti izgled elektronskog programskog vodiča u grafičkoj korisničkoj sprezi.

Slika 15: Prikaz elektronskog programskog vodiča

Verifikacija elekronskog programskog vodiča je urađena na sledeći način. Procesom pretrage pronađeni su OTT servisi i DTT servisi sa obe frekvencije.

Elekronski programski vodič za OTT servise je verifikovan upoređivanjem dobijenih rezultata sa rezultatima dobijenim slanjem zahteva mrežnom poslužiocu preko Postman aplikacije.

Za potrebe verifikacije DTT elekronskog programskog vodiča potreban je komercijalni uređaj koji ima mogućnosti DTT pretrage servisa i koji treba povezati na testnu lokalnu DTT mrežu. Početak testa zahteva reprodukciju istog servisa na oba uređaju. Kada se elekronski programski vodič prikupi pristupa se upoređivanju sadržaja na oba uređaja za sedam dana za taj servis i za ostale servise sa iste frekvencije. Potom se vrši promena servisa na servis sa druge frekvencije i čeka se dobavljanje elekronskog programskog vodiča. Po završetku dobavljanja upoređuje se sadržaj na oba uređaja za sve servise sa te frekvencije za period od sedam dana. Test je uspešno izvršen zato što se sadržaj elekronskog programskog vodiča na oba uređaja poklapa za sve servise.

6. Zaključak

Ovim radom prikazano je jedno rešenje proširenja Android operativnog sistema integracijom postojeće programske podrške za digitalnu televiziju na hibridnom STB uređaju. U doba razvoja mrežne infrastrukture i porasta informacione i tehnološke pismenosti porasla je potreba za korišćenje STB uređaja koji bi pored televizijskog sadržaja nudio i sadržaje sa mreže.

U ovom radu je realizovana programska podrška pretrage digitalnih mrežnih i televizijskih servisa, dobavljanje elektronskog programskog vodiča i komunikacija uređaja sa mrežnim poslužiocem.

U okviru rada uvedeno je više slojeva programske podrške. Rešenje je izvedeno tako da grafička korisnička sprega ne sadrži nikakvu logiku za pretragu servisa, dobavljanje elektronskog programskog vodiča i komunikaciju sa mrežnim poslužiocem, već samo poziva funkcije nižih slojeva i zahteva izvršenje određenih radnji. Na ovaj način moguće je lako zameniti grafičku korisničku spregu sa nekom drugom bez mnogo napora i utroška vremena.

Rešenje je verifikovano nad opisanim skupom testova i dobijeni rezultati su upoređivani sa komercijalnim uređajima i pokazali su da nema odstupanja pri pretrazi kanala i prikazu elektronskog programskog vodiča. Prikaz OTT sadržaja sa mrežnog poslužioca verifikovan je pomoću Postman aplikacije upoređivanjem odgovora od mrežnog poslužioca i prikazanog sadržaja u grafičkoj korisničkoj sprezi.

Realizacijom ovakvog sistema i demonstracijom njegovih mogućnosti, uočava se i postojanje prostora za dalji napredak. Postoje čak tri pravca koja se mogu pratiti u daljem razvoju programske podrške:

1. uvođenjem novih API funkcija u MAL sloju mogle bi se dobiti sledeće DTT funkcionalnosti: teletekst, prevode, snimanje sadržaja itd.

2. proširenjem programske podrške za OTT sadržaje kao što su Catch Up, VOD itd., kao i uvođenjem servisa koji se naplaćuju (npr. sportski kanali)
3. kombinacija DTT i OTT sadržaja na sledeći način: ako se na mrežnom poslužiocu i u DTT mreži nalaze isti servisi onda se može koristiti elektronski programski vodič samo sa mrežnog poslužioca (u slučaju da je ažurniji) ili u slučaju lošeg DTT signala da se servis automatski reprodukuje kao OTT servis...

7. Literatura

[1] Wikipedia: https://en.wikipedia.org/wiki/Digital_television , učitano 24.05.2015.

[2] ETSI EN 302 307 V1.1.1 (2004-06), European Standard (Telecommunications series), Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, Gathering and other broadband satellite applications

[3] Hervé Benoît: Digital Television: MPEG-1,MPEG-2 and Principles of the DVB System, Taylor & Francis, 2002

[4] Predavanja iz predmeta Projektovanje namenskih računarskih struktura 1: http://www.rt-rk.uns.ac.rs/studijski-program-2009/vi-2009/pnrs1/737-pnrs1-predavanja , učitano 06.07.2015

[5] ETSI EN 300 707, Electronic Programme Guide (EPG); Protocol for a TV Guide using electronic data transmission

[6] Nemanja Lukić, "Predlog proširenja Android operativnog sistema servisima digitalne televizije", doktorska disertacija, Fakultet Tehničkih Nauka, Novi Sad 2014

[7] Whitepaper: Over The Top TV (OTT TV) Platform Technologies, Endurance Technology, november 2009

[8] M. Vidakovic, N. Teslic, T. Maruna, V. Mihic, "Android4TV: a proposal for integration of DTV in Android devices," 30th IEEE International Conference on Consumer Electronics (ICCE), January 2012, pp. 441-442.

[9] Walter Fischer, “Digital Video and Audio Broadcasting Technology”, Springer Science & Business Media, 2010