UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA U NOVOM SADU

Ilija Đukić

Jedno rešenje softvera za integraciju pretrage TV baziranih podataka u uređajima zasnovanim na Android OS

MASTER RAD

Novi Sad, 2014

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**:		Илија Ђукић	
Ментор, **МН**:		проф. др Никола Теслић	
Наслов рада, **НР**:		Једно решење софтвера за интеграцију претраге ТВ базираних података у уређајима заснованим на Андроид ОС	
Језик публикације, **ЈП**:		Српски / латиница	
Језик извода, **ЈИ**:		Српски	
Земља публиковања, **ЗП**:		Република Србија	
Уже географско подручје, **УГП**:		Војводина	
Година, **ГО**:		2014	
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)		8/74/0/9/26/0/0	
Научна област, **НО**:		Електротехника и рачунарство	
Научна дисциплина, **НД**:		Рачунарска техника	
Предметна одредница/Кључне речи, **ПО**:		Андроид оперативни систем, дигитална телевизија, Јава програмски језик, претрага	
УДК			
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад	
Важна напомена, **ВН**:			
Извод, **ИЗ**:		Овај рад се бави истраживањем могуће интеграције ТВ базиране претраге у Андроид оперативни систем. Циљ рада је да дефинише програмску спрегу којом је могуће извршавати претрагу ТВ базираних података из апликативног домена у Андроид ОС. Имплементација саме претраге се ограничава на податке везане за ЕПГ, телетекст и ПВР информације.	
Датум прихватања теме, **ДП**:			
Датум одбране, **ДО**:			
Чланови комисије, **КО**:	Председник:	Иштван Папп	
	Члан:	Никола Челановић	Потпис ментора
	Члан, ментор:	Никола Теслић	

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**:		Ilija Djukic	
Mentor, **MN**:		Nikola Teslic, PhD	
Title, **TI**:		One software solution for search integration of TV based data in Android OS based devices	
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**:		2014	
Publisher, **PB**:		Author's reprint	
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6	
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		8/74/0/9/26/0/0	
Scientific field, **SF**:		Electrical Engineering	
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, **S/KW**:		Android operating system, digital television, Java programming language, search	
UC			
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, **N**:			
Abstract, **AB**:		Purpose of this paper is to research the possibility of integration of TV based search functionality in Android operating system. It's goal is to define application programming interface with TV based search for Android application development. Implementation of this search functionality is bounded to EPG, Teletext and PVR information.	
Accepted by the Scientific Board on, **ASB**:			
Defended on, **DE**:			
Defended Board, **DB**:	President:	Istvan Papp	
	Member:	Nikola Celanovic	Menthor's sign
	Member, Mentor:	Nikola Teslic	

Zahvalnost

Želeo bih da se zahvalim prof. dr Nikoli Tesliću na podsticanju i iskrenoj podršci tokom upisa i za vreme master studija.

Takođe bih se zahvalio i Nemanji Lukiću na kreativnim savetima i strpljivoj saradnji čiji je rezultat ovaj master rad.

SADRŽAJ

SPISAK SLIKA

SPISAK TABELA

SKRAĆENICE

API - *Application Programming Interface*, aplikativna programska sprega

CPU - *Central Processing Unit*, centralna procesorska jedinica

DSH - *Data Structure Handler*, rukovalac strukturama podataka

DTV - *Digital Television*, digitalna televizija

DVB - *Digital Video Broadcast*, emitovanje digitalnog video signala

DVB-C - *Digital Video Broadcast – Cable*, kablovsko emitovanje digitalnog video signala

DVB-S - *Digital Video Broadcast – Satellite*, satelitsko emitovanje digitalnog video signala

DVB-T - *Digital Video Broadcast – Terrestrial*, zemaljsko emitovanje digitalnog video signala

DVR - *Digital Video Recorder*, digitalni video snimač

EPG - *Electronic Program Guide*, elektronski programski vodič

GPU - *Graphical Processing Unit*, grafička procesorska jedinica

HbbTV - *Hybrid Broadcast Broadband TV*, Hibridna televizija

HD - *High Definition*, visoka definicija (kvalitet)

IPC - *Inter Process Communication*, Među-procesna komunikacija

JAR - *Java Archive*, Java statička biblioteka

JNI - *Java Native Interface*, Java izvršna sprega

MAL - *Middleware Abstraction Layer*, sloj apstrakcije sistemske podrške

MHEG - *Multimedia and Hypermedia Experts Group*, ekspertska grupa za multimediju i hipermediju

MW - *Middleware*, sistemska podrška

OS - *Operating System*, operativni sistem

PES - *Packetized Elementary Stream*, paket elementarnog prenosa

PSI - *Program Specific Information*, informacije specifične za program

PVR - *Personal Video Recorder*, personalni video snimač

QSB - *Quick Search Box*, polje za brzu pretragu

RAM - *Random Access Memory*, memorija sa slučajnim pristupom

RT - *Real time*, u realnom vremenu

SDK - *Software Development Kit*, softverski razvojni alat

SoC - *System on Chip*, integrisano kolo fizičke arhitekture

STB - *Set Top-Box*, digitalni televizijski prijemnik

TV - *Television*, televizija

URI - *Uniform Resource Identifier*, jedinstveni identifikator izvora informacije

VM - *Virtual Machine*, virtuelna mašina

VOD - *Video On Demand*, video na zahtev

1. Uvod

Od nekada relativno jednostavnih uređaja za praćenje video sadržava, TV aparati postaju sve kompleksnije (pametnije) mašine i pružaju korisniku mnoštvo usluga, kao što je pristup servisima za iznajmljivanje i kupovinu filmova preko interneta (engl. Video On Demand, ili VOD services), pregled internet sadržaja i pristup internet socijalnim mrežama. Danas se u TV uređaje ugrađuju savremeni operativni sistemi sa mogućnošću paralelnog izvršavanja više programa i ovi uređaji, po tehničkim karakteristikama, sve više počinju da liče na personalne računare.

Poslednji trend u izbora operativnog sistema za TV uređaj je integracija Android operativnog sistema. Android je operativni sistem otvorenog izvornog koda, zasnovan na Linux jezgru, koji sve više preplavljuje tržište mobilnih telefona i tabličnih računara. Ovaj operativni sistem raspolaže ogromnom bazom aplikacija i njegovi tvorci ga neprekidno dopunjuju novim proširenjima.

Korisnici TV uređaja su obično navikli na jednostavan način upravljanja ovim aparatom, a kako se proširuju njegove mogućnosti povećava se i količina podataka dostupnih na tom uređaju, tako da pretraga za određenim informacijama može biti naporna krajnjem korisniku. Uzevši u obzir povećan broj podataka i korisničke navike, u ovakvom okruženju je neophodno postojanje mogućnosti za brzu pretragu informacija dostupnim na TV uređaju.

U ovom radu će biti predloženo jedno rešenje za implementaciju pretrage različitog sadržaja vezanog za digitalnu televiziju na uređajima zasnovanim na Android operativnom sistemu. Cilj ovog rada je sledeći:

- Definisati aplikativne programsku spregu (API) kojom se lako može doći do rezultata pretrage i koja će biti nezavisna od implementacije same pretrage.

- Implementirati mehanizam za prikupljanje dostupnih TV podataka i formiranje baze pretrage, do čijih se rezultata dolazi primenom definisane aplikativne programske sprege.
- Integrisati pretragu TV sadržaja u Android sistemsku aplikaciju za brzu pretragu (QSB) pomoću definisane aplikativne programske sprege.
- Obezbediti korisniku mogućnost adekvatnog prikaza odabranih rezultata pretrage. Ovo podrazumeva da u Android sistemu postoji aplikacija koja prihvata QSB rezultate i shodno tome preduzima odgovarajuće akcije.

Za realizaciju ovog rešenja korišćen je Java programski jezik (koji je i primarni programski jezik za razvoj aplikacija u Android OS), Android alat za razvoj softvera (SDK) i programska podrška za razvoj TV aplikacija u Android operativnom sistemu – Android4TV SDK [Vidakovic].

Ovaj rad je sačinjen od sedam poglavlja.

U poglavlju 2 je dat kratak pregled razvoja oblasti digitalne televizije i opisana je uloga Android operativnog sistema u ovoj oblasti.

Poglavlje 3 se bavi istraživanjem dosadašnjih dostignuća u oblasti istraživanja ovog rada. U ovom poglavlju su navedeni naučni radovi koji predstavljaju osnovu za istraživanja ovog rada i čija su rešenja ovde usvojena i proširena u cilju daljeg istraživanja oblasti digitalne televizije.

Poglavlje 4 opisuje mere koje se koriste za ocenu kvaliteta i funkcionalnosti implementiranog sistema.

U poglavlju 5, dat je detaljan opis implementacije softverskog rešenja za pretragu TV relevantnih podataka u uređaju baziranom na Android operativnom sistemu. U ovom poglavlju su takođe predstavljena sistemska proširenja usvojenih rešenja za TV podršku na Android platformi kao i opis softverske arhitekture kompletnog rešenja.

Poglavlje 6 se bavi rezultatima merenja ranije opisanim metrikama za ocenu kvaliteta predloženog rešenja. Na kraju poglavlja, izložena je uporedna analiza realizovanog rešenja sa postojećim rešenjima.

U poglavlju 7, dat je zaključak istraživanja sa posebnim osvrtom na pravce daljeg razvoja.

2. Pregled razvoja oblasti istraživanja i postavka ciljeva istraživanja

Digitalna televizija (engl. Digital television, DTV) je oblast potrošačke elektronike koja se bavi uređajima čija je primarna namena prijem digitalnog (obično i analognog) TV signala i prezentacijom video i audio sadržaja. Ovo je oblast potrošačke elektronike koja mora zadovoljiti mnoge tradicionalne, i istorijske, zahteve kako samih potrošača tako i propisanih standarda. Međutim i pored grubih okvira ove delatnosti, ovde postoji i dosta prostora za inovaciju i uvođenje novih tehnologija u navike potrošača (korisnika).

2.1 Digitalna TV tehnologija

Najveća tehnološka evoluciju na polju TV prenosa slike i zvuka, nakon prelaska sa crno-bele na kolor tehniku, predstavlja prelazak sa analognog da digitalni signal. Prelazak sa analognog na digitalni TV signal je, prevashodno, posledica potrebe za poboljšanjem kvaliteta TV slike. Trenutno postoje četiri različita standarda emitovanja zemaljskog digitalnog TV signala:

- ATSC (engl. Advanced Television System Committee) je standard koji je razvijen početkom devedesetih godina u Sjedinjenim Američkim Državama pod pokroviteljstvom konzorcijuma za elektroniku i telekomunikacionih kompanija, prevashodno za potrebe prenosa slike visoke rezolucije. Države koje su prihvatile ovaj standard su Sjedinjene Američke Države, Kanada, Meksiko, Južna Koreja, Dominikanska Republika i Honduras.
- DVB-T (engl. Digital Video Broadcasting-Terrestrial) je standard koji je prvi put objavljen 1997 godine a prvi put je emitovan u Velikoj Britaniji 1998 godine. Danas ovaj standard pripada porodici DVB standarda koja pokriva standarde za

prenos slike putem zemaljskog signala (DVB-T), satelitskog signala (DVB-S) i pute kabla (DVB-C). Ovaj standard je prihvaćen u Evropi, Australiji i na Novom Zelandu.

- ISDB-T (engl. Terrestrial Integrated Services Digital Broadcasting) je standard čiji je razvoj počeo još šezdesetih godina u Japanu i koji je inspirisao Sjedinjene Američke Države da porade na sopstvenom standardu za prenos slike visoke rezolucije. Ovaj standard je usvojen u Japanu i na Filipinima. Takođe postoji i internacionalna adaptacija ovog standarda (engl. ISDB-T International) koja se koristi u mnogim Južno-Američkim zemljama kao i u Afričkim zemljama portugalskog govornog područja.
- DTMB (engl. Digital Terrestrial Multimedia Broadcasting) standard je zvanično objavljen 2002 godine za potrebe emitovanja digitalnog TV signala u Narodnoj Republici Kini, uključujući Hong Kong i Makao.

Postoje mnoge prednosti prelaska sa analognog TV signala na digitalni, a najznačajnija je ta da prenos digitalnih kanala zahteva manji propusni opseg u frekventnom spektaru, tako da se količina prenesenih podataka mnogostruko povećava. Prenos veće količine podataka je prevashodno bitan za mogućnost emitovanja slike visokog kvaliteta (HD), a takođe omogućava i emitovanje više digitalnih kanala na istom frekventnom opsegu (na kojem se mogao emitovati samo jedan analogni kanal).

Slika 2.1 Poređenje kvaliteta slike, analogna (levo) i digitalna (desno) tehnologija

Tehnika „pakovanja“ više digitalnih kanala naziva se multipleksiranje. Takođe, digitalni TV signal omogućava i prenošenje sadržaja različite vrste (pored slike i zvuka) kao što su podrška za više jezika i elektronski programski vodič. Ovi dodatni sadržaji su posebno važni za ovaj rad, o čemu će se više govoriti u nastavku teksta.

2.1.1 Interaktivni sadržaji digitalne televizije

Prelaskom sa analognog na digitalni TV signal otvaraju se nove mogućnosti interakcije korisnika sa TV uređajem. Pored slike i zvuka, digitalnim signalom je moguće prenositi različite sadržaje informativnog karaktera koje proširuju upotrebne mogućnosti TV uređaja. Emitovanje ovakvih sadržaja je definisano različitim standardima.

Teletekst (engl. Teletext) je informativni servis kreiran u Velikoj Britaniji početkom 1970-tih godina. Teletekst servisom se formira mozaik slika na televizijskom ekranu koristeći tekst i jednostavne geometrijske oblike. Ovako formirane slike se nazivaju teletekst stranice i na jednom TV programu je moguće imati do 800 ovakvih stranica kojima korisnik može da pristupi. Ovim stranicama se obično prenose vesti, razne informacije vezane za TV program i različiti informativni i zabavni sadržaji.

Slika 2.2 Prikaz jedne teletekst stranice

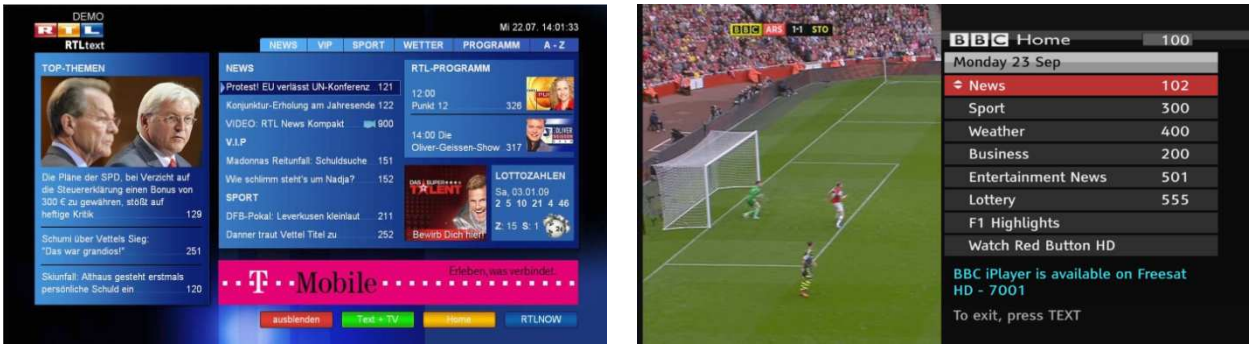
EPG (engl. Electronic programming guide), ili elektronski programski vodič, je informativni televizijski servis kojim se prenose informacije o tekućim i budućim emisijama koje se emituju preko televizijskih ili radio programa. Servisom se prenose informacije o nazivima i sadržajima emisija, kao i vremenski raspored njihovog emitovanja. Za razliku od teletekst servisa, EPG prenosi samo informativne podatke, ne i način njihovog prikazivanja, tako da je sama implementacija programa koji će prezentovati EPG podatke ostavljena proizvođačima TV uređaja.

Teletekst i EPG su najviše rasprostranjeni i najčešće korišćeni informativni TV servisi u Evropi. Takođe treba napomenuti da teletekst standard postoji i u analognom TV signalu, međutim zbog popularnost ovog servisa on nastavlja da postoji i u digitalnoj televiziji.

Slika 2.3 Primer jednog grafičkog prikaza EPG tabele

Pored navedenih postoje i mnogi drugi TV servisi čija rasprostranjenost zavisi od regiona do regiona. Interaktivni TV servisi koji su takođe veoma popularni u Evropi su:

- HbbTV (engl. Hybrid Broadcast Broadband Television) je interaktivni TV servis, sličan teletekst servisu, koji kombinuje internet sadržaj sa trenutno gledanim TV programom. Najpopularniji je u Nemačkoj, Francuskoj, Španiji i ostalim zemljama Evrope.
- MHEG-5 (engl. Multimedia and Hypermedia Experts Group) je besplatan i javno dostupan standard za prenos interaktivnog TV sadržaja. On pokriva širok spektar interaktivnih TV servisa. Najzastupljeniji je u Velikoj Britaniji, Novom Zelandu, Hong Kongu i Južno Afričkoj Republici.



Slika 2.4 Primeri interaktivnih TV servisa, HbbTV (levo) i MHEG-5 (desno)

Digitalna televizija takođe omogućava i prenošenje sadržaja u vidu višejezične podrške (višestruki audio TV servisi) kao i podršku za tekstualne pod-natpise (engl. Subtitles).

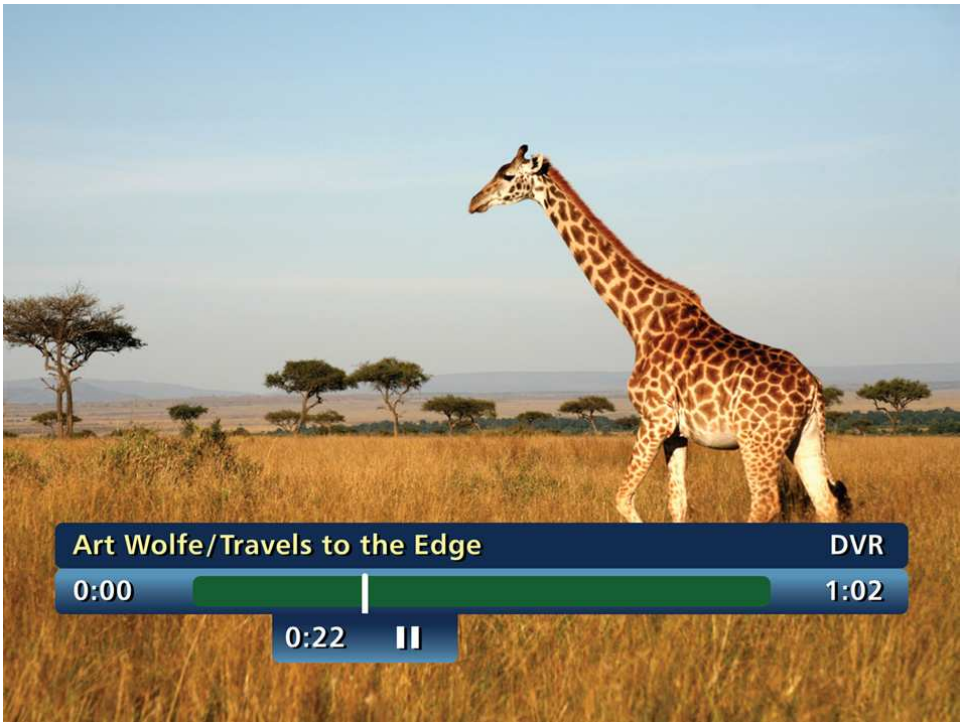
2.1.2 Snimanje digitalnog video sadržaja

Još jedna karakteristika TV sistema koja je bitno promenjena prelaskom sa analogne na digitalnu tehnologiju je snimanje TV sadržaja. U analognoj tehnologiji se snimanje audio i video

sadržaja beležilo na magnetne medije (video kasete) a zbog analogne prirode zapisa kvalitet samog snimka je zavisio od mnogih faktora (kvalitet hardverskih komponenti u sistemu za snimanje, kvalitet samih magnetnih medija i razne vrste interferencije signala) i gotovo nikada nije u potpunosti odgovarao izvornom sadržaju.

Prelaskom na digitalnu tehnologiju snimanje audio i video sadržaja je znatno unapređeno. Prednosti digitalnog video snimača (DVR) su sledeće:

- Snimljeni sadržaj je identičan kao i izvorni materijal.
- Implementacija samog snimanja je softverska i svaki uređaj koji prijemnik digitalnog TV signala može biti i snimač.
- Za beleženje snimljenog sadržaja se mogu koristiti široko rasprostranjeni memorijski medijumi.



Slika 2.5 Primer grafičkog prikaza digitalnog TV snimača

Snimanje digitalnog TV sadržaja je često u tesnoj vezi sa EPG servisom i u komercijalnim TV rešenjima su ove dve tehnike obično integrisane u jedinstven sistem poznat pod nazivom personalni video snimač (PVR). U PVR sistemu je korisniku TV uređaja omogućeno da pomoću EPG tabelarnog prikaza označi emisije u budućnosti koje trebaju biti snimljene. Na taj način korisnik ne mora ručno da startuje i stopira snimanje već će to PVR sistem uraditi za označene emisije. Postoje i implementacije PVR sistema koje idu korak dalje i integrišu algoritme za praćenje navika korisnika i automatski snimaju emisije koje bi se korisniku svidele.

Još jedna primena tehnike digitalnog snimanja je u implementaciji sistema za odloženo gledanje (engl. Time Shift). Ovaj sistem omogućava korisniku TV uređaja da u svakom trenutku može da pauzira program koji uživo gleda i nastavi sa gledanjem kasnije.

2.2 Karakteristike Android OS-a

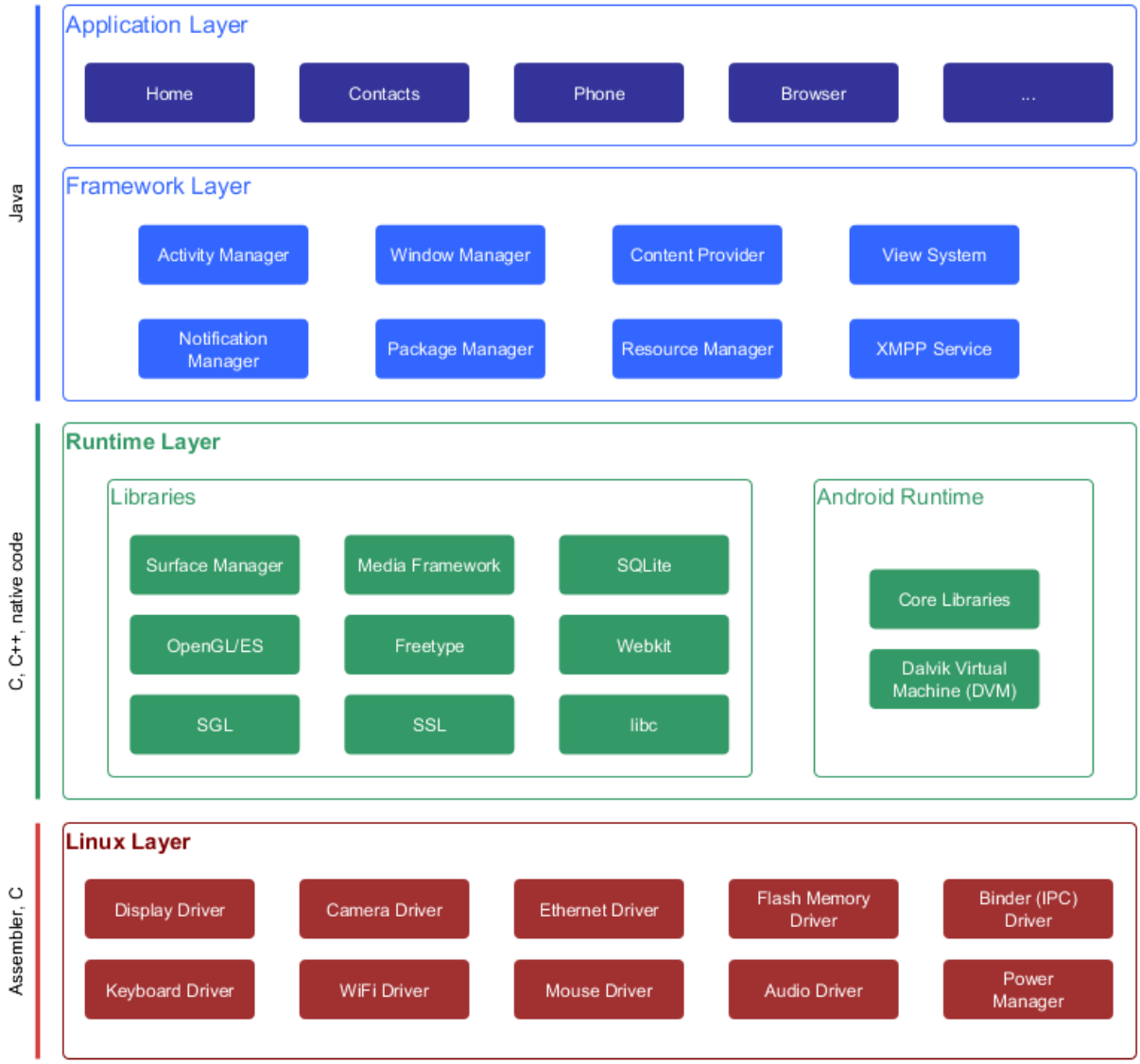
Android je trenutno jedan od najzastupljenijih operativnih sistema na polju mobilnih i tabličnih uređaja. Android je operativni sistem otvorenog koda, baziran na Linux jezgru, koji je razvijen od strane The Open Handset Alliance (grupa od preko 30 kompanija predvođenih kompanijom *Google*). Android je prvi put najavljen 2007. godine a prvi komercijalno raspoloživi telefon, praćen i objavljivanjem prve verzije operativnog sistema, pojavio se na tržištu krajem 2008. godine. Osnovne karakteristike ovog operativnog sistema su:

- Aplikativno radno okruženje: Aplikativno okruženje se koristi od strane autora aplikacija prilikom razvoja aplikacija za ovaj operativni sistem (Android aplikacije). Ovo okruženje je javno dostupno i izuzetno detaljno dokumentovano [Ableson].
- *Dalvik* virtuelna mašina: Android OS poseduje svoju implementaciju Java virtuelne mašine (ne oslanja se na standardnu Sun Java VM). *Dalvik* interpretira Java bajt-kod generisan od strane Java prevodioca.
- Integrisan Internet pretraživač: Android uključuje internet pretraživač baziran na [WebKit] kao deo standardne liste Android aplikacija. Ovaj pretraživač je na raspolaganju autorima Android aplikacija kroz standardne komponente aplikativnog radnog okruženja.
- Optimizovana grafička sprega: Android u sebi sadrži podršku kako za 2D tako i 3D napredne grafičke operacije. Prikazivanje grafike je podržano od strane hardvera OpenGL ES standardom [OpenGL|ES].
- SQLite: Standardna implementacija sprege baza podataka na raspolaganju autorima Android aplikacija kroz standardne komponente aplikativnog radnog okruženja.
- Mrežne sprege: Android uključuje podršku za većinu bežičnih tehnologija, kao što su Bluetooth, EDGE, i 3G.
- Bogato razvojno okruženje: Razvojno okruženje (engl. Software Development Kit, SDK) je besplatno i dostupno autorima, i uključuje emulator i alate za uklanjanje grešaka i analizu programskog koda [Android SDK].

Za razvoj aplikacija za Android operativni sistem, koristi se Java programski jezik, a aplikacije se izvršavaju u pomenutoj Dalvik virtuelnoj mašini. Pošto je Android operativni sistem koji se ugrađuje u različite uređaje, zasnovane na različitim platformama (hardverskim

arhitekturama), izbor da se aplikacije izvršavaju u virtuelnoj mašini je neizbežan da bi se održala portabilnost aplikacija na različitim uređajima.

Glavni razlog integracije virtuelne mašine u ugrađenim sistemima je u vezi sa programskim jezikom koji je odabran za aplikativni razvoj. U ugrađenim sistemima, ne postoji jedan programski jezik koji je savršeno rešenje za svaki sistem, odnosno platformu. Mnogi kompleksni ugrađeni sistemi sadrže višestruke nivoe programske podrške pisanih u različitim programskim jezicima. Ovo je slučaj i sa Android operativnim sistemom, gde je nivo sistemskih servisa pisan u asemblerskom i C programskom jeziku, sistemski nivo u C i C++, dok su nivoi aplikativnog radnog okruženja i samih aplikacija pisani u Java programskom jeziku. Dijagram [Slika 2.6] prikazuje različite slojeve programske podrške Android operativnog sistema, sa posebnim osvrtom na korišćene programske jezike prilikom realizacije tih slojeva.



Slika 2.6 Softverska arhitektura Android operativnog sistema

Osim ovih karakteristika, Android operativni sistem poseduje nekoliko osobina koje ga čine posebno interesantnim kao izbor platforme za ugrađene sisteme [Lukic]:

- Širok ekosistem aplikacija: U vreme pisanja ovog rada Android market broji preko 1 000 000 raspoloživih aplikacija
- Konzistentni aplikativni API: Sve programske sprege koje pruža aplikativno radno okruženje su namenjene tako da budu unapred-kompatibilne. Drugim rečima, sve prethodno razvijene Android aplikacije će nastaviti da funkcionišu normalno i u budućim verzijama operativnog sistema.
- Zamenljivost komponenti: Pošto je Android operativni sistem otvorenog tipa, većinu podrazumevanih komponenti je moguće promeniti. Primeri za ovaj princip su zamena osnovne aplikacije (engl. Home-screen launcher) ili podrazumevanog multimedijalnog radnog okruženja (engl. Stagefright)
- Proširivost: Druga prednost otvorenosti Android operativnog sistema je mogućnost proširenja karakteristika platforme i dodavanja podrške za nove komponente fizičke arhitekture. Ovo je moguće korišćenjem Android HAL (engl. Hardware Abstraction Layer) nivoa.

2.2.1 Internet pretraga u Android OS

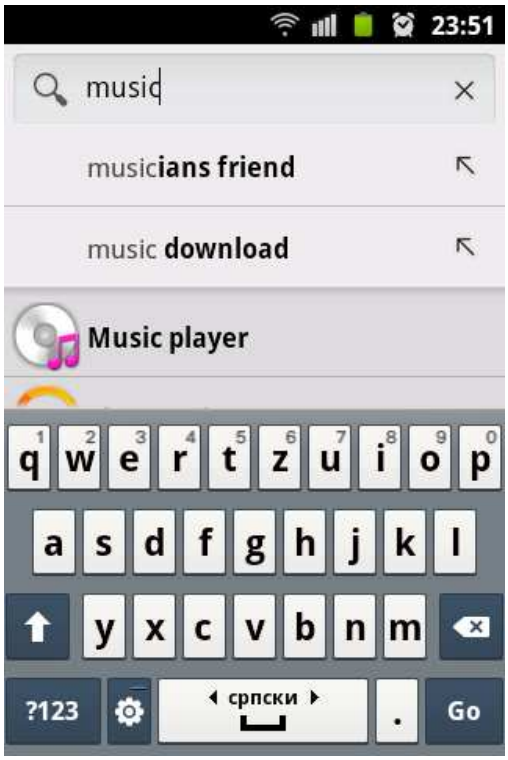
Jedna od karakteristika Android operativnog sistema je integrisana pretraga internet sadržaja koja je izuzetno jednostavna za korišćenje. Može se reći da se ovakva funkcionalnost i očekuje od operativnog sistema koji je razvila kompanija Google, čiji je primarni vid poslovanja internet pretraga.

Sistem za pretragu u Android OS je takođe moguće proširiti i omogućiti mu da osim pretrage interneta, kroz isti mehanizam za pretragu korisniku ponudi i neke dodatne rezultate pretrage. Sistem za pretragu je moguće proširiti kroz aplikativni nivo i korišćenjem standardnog (javnog) Android okruženja za razvoj aplikacija (Android SDK). S toga je razlog izbora ovog operativnog sistema kao bazne platforme za predmet istraživanja ovog rada upravo integrisana pretraga u Android OS.

2.2.2 Aplikacija za brzu pretragu u Android OS

Svako Android okruženje podrazumeva da je korisniku dostupna sistemska aplikacija za brzu pretragu (engl. Quick Search Box, QSB). Sa stanovišta korisnika Android operativnog sistema, ova aplikacija predstavlja polje za unos teksta pretrage. Nakon zahteva za pretragom aplikacija će korisniku predstaviti spisak rezultata relevantnih za dati uređaj. Sledi par primera mogućih rezultata pretrage korišćenjem QSB aplikacije:

- Prečice ka najčešće korišćenim aplikacijama koje su instalirane na dati uređaj.
- Ako je uređaj povezan na internet mrežu lista rezultata može sadržati prečice ka internet sajtovima koje je pronašao Google pretraživač ili video snimke koje je pronašao internet servis YouTube.
- Ako se radi o mobilnom telefonu lista rezultata može sadržati kontakte osoba iz telefonskog imenika.
- Ako uređaj podržava GPS konekciju lista rezultata može sadržati lokacije najbližih objekata od značaja.



Slika 2.7 Prikaz brze pretrage na standardnom Android telefonu

Za dobavljanje rezultata pretrage QSB aplikacija se oslanja na posebne programske ekstenzije zvane dobavljači podataka (engl. Search Provider, u daljem tekstu SP). Dobavljači podataka mogu biti deo bilo koje Android aplikacije, oni se posebnim funkcijama registruju kao globalni dobavljači podataka i od tog trenutka svaki zahtev za pretragu nad QSB aplikacijom će zahtev proslediti ka dobavljačima podataka. Prethodno navedeni primeri pretrage predstavljaju osnovne dobavljače podataka koji se obično nalaze u svakom Android uređaju, međutim ovo ipak zavisi od proizvođača uređaja.

2.3 Digitalna televizija i Android OS

Operativni sistemi koji se ugrađuju u TV uređaje (i druge uređaje potrošačke elektronike) se nazivaju „ugradni sistemi“ (engl. Embedded Systems) i obično se baziraju na Linux jezgru. Tržištem ugradnih sistema su dugo vladali razni sistemi „u realnom vremenu“ (RT), u glavnom

varijante Linux distribucija. Prioritet ovih sistema je prevashodno bio da uređaju obezbedi nesmetanu osnovnu namensku funkcionalnost uz minimalna opterećenja samog hardvera.

Današnji TV uređaji su odavno prestali da budu samo aparati za praćenje TV programa. Tako se danas od TV uređaja očekuje da podržava različite napredne funkcije, kao što su:

- prikaz različitog interaktivnog TV sadržaja,
- reprodukcija multimedijalnog sadržaja (kao što su video, audio, statičke slike, itd...) sa različitih izvora (USB konekcije, lokalna i internet mreža, itd...),
- pristup internetu i internet multimedijalnim i informativnim servisima,
- komunikacija sa prijateljima putem socijalnih mreža i različitih VOPI servisa.

Da bi se ovo sve postiglo, u TV uređaj se mora ugraditi operativni sistem koji će biti u mogućnosti da izvršava programe koji omogućuju ove sadržaje. Kako su napredovale hardverske karakteristike samih TV uređaja (što je prevashodno vezano za napredak tehnologije izrade jeftinijih procesorskih jedinica) menjala se i kompleksnost samih operativnih sistema za TV uređaje. Najnoviji trend u razvoju TV uređaja je ugradnja Android operativnog sistema u ovaj uređaj.

2.3.1 Okruženje za razvoj TV aplikacija u Android OS

Iako veoma pogodan kao TV platforma, Android je operativni sistem prevashodno namenjen prenosivim uređajima sa ekranom osetljivim na dodir, tako da ne postoji integrisana podrška za razvoj TV aplikacija. Integracija Android OS u TV i STB platforme je predmet mnogih istraživanja i postoje mnoga komercijalna rešenja, međutim ova rešenja su u glavnom zatvorenog tipa i nisu namenjena otvorenom razvoju softvera.

Kao osnova za razvoj softverske TV podrške, u ovom radu je izabrano otvoreno rešenje [Android4TV].

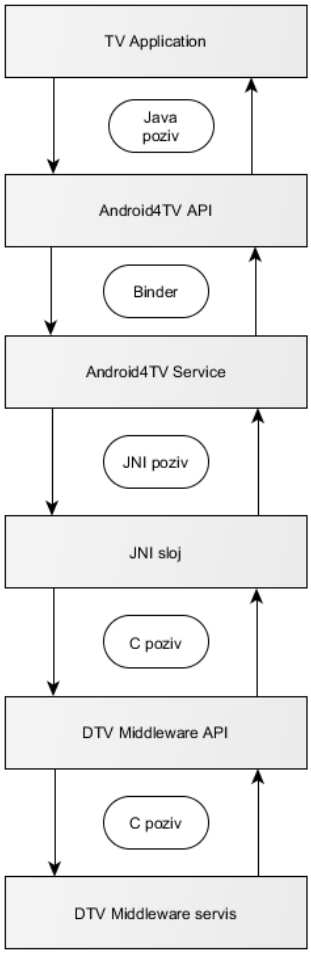
2.3.2 Android4TV razvojno okruženje

Android4TV razvojno okruženje (engl. Framework) je softversko rešenje, razvijeno od strane kompanije iWedia, koje pruža podršku za punu TV funkcionalnost prilikom razvoja aplikacija u Android sistemu. Ovo okruženje nudi širok dijapazon mogućnosti i pokriva gotovo sve aspekte razvoja TV softvera kroz definisanu aplikativnu programsku spregu, Android4TV API.

Pored aplikativne programske podrške (API) u okviru Android4TV okruženja je i Android4TV aplikacija, koja predstavlja kompletno aplikativno grafičko rešenje za interakciju krajnjeg korisnika sa TV uređajem. Ova aplikacija je otvorenog koda, i može se preuzeti sa [A4TV App].

Slika 2.8 Izgled Android4TV aplikacije

Struktura Android4TV razvojnog okruženja sastoji se iz više programskih slojeva, što se može videti na dijagramu [Slika 2.9].



Slika 2.9 Softverska arhitektura Android4TV razvojnog okruženja

Sa priloženog dijagrama se može videti da gornji sloj softverske podrške čine Android aplikacija (Android4TV Application), koja predstavlja kompletno rešenje za korišćenje Android platforme kao TV uređaja, sa stanovišta korisnika. Ova aplikacija je ujedno i primer korišćenja Android4TV API-ja za izradu TV baziranih aplikacija na Android OS-u.

Android4TV aplikacija se oslanja na pomenuti Android4TV API sloj, koji apstraktuje TV funkcionalnost i grupiše je u Java klasne programske sprege (engl. Java Interface). Da bi bilo koja Android aplikacija koristila Android4TV API, neophodno je da se poveže sa statičkom Java bibliotekom (JAR) – *android4tv-framework.jar*. Ova Java biblioteka sadrži sve komponente Android4TV API Java kasnih sprega i može se besplatno preuzeti sa [Android4TV].

Android4TV Service predstavlja Android Java aplikativni servis i u njemu su implementirane klase Android4TV API sloja. Aplikacija koja koristi Android4TV API će indirektno komunicirati sa ovim slojem. Ova komunikacija se izvodi mehanizmom za komunikaciju među programskim procesima (IPC) u Android OS koji se naziva Binder.

JNI sloj predstavlja mehanizam povezivanja Java i izvršnog (engl. Native) domena (C, C++) koji je tipičan za Java programiranje. Ovaj sloj implementira generičko preslikavanje funkcija iz DTV Middleware API sloj u Android4TV API.

DTV Middleware API se najjednostavnije može opisati kao Android4TV API za mašinski domen. Ovaj sloj izdvaja funkcije TV sistemske programske podrške (TV MW) i omogućava programima pisanim za mašinski domen, u C jeziku, da pristupe TV funkcijama uređaja. Ovaj sloj se još naziva i apstrakcioni među sloj za TV sistemsku programsku podršku (MAL).

DTV Middleware service sloj predstavlja implementaciju same TV sistemske podrške. Ovaj sloj predstavlja kompleksan softver koji korišćenjem sistemskih i hardverskih komponenti Android uređaja formira jedinstvenu softversku celinu za procesiranje svih aspekata digitalne televizije.

U ovom radu će fokus biti samo na komponentama implementiranim u aplikativnom Java domenu, a to su:

- Android4TV API
- Android4TV Service
- Android4TV Application

2.4 Postavka ciljeva istraživanja

Cilj ovog rada je da predstavi rešenje softverske implementacije koja omogućava pretragu TV baziranih podataka na TV uređaju zasnovanom na Android platformi, kroz mehanizam standardne pretrage u ovom operativnom sistemu.

Realizacija ovog rešenja je (kao što je već rečeno u uvodu) podeljena u tri zadatka:

1. Definisanje aplikativne programske podrške kojom se lako može doći do rezultata pretrage i koja će biti nezavisna od implementacije same pretrage.
2. Realizacija same pretrage dostupnih TV podataka, do čijeg se rezultata dolazi primenom definisane aplikativne programske podrške.
3. Integrisanje pretrage TV sadržaja u Android sistemsku aplikaciju za brzu pretragu (QSB) pomoću definisane aplikativne programske podrške.
4. Pored navedenih ciljeva neophodno je i obezbediti podršku sistema za reakciju na odabrane rezultate pretrage.

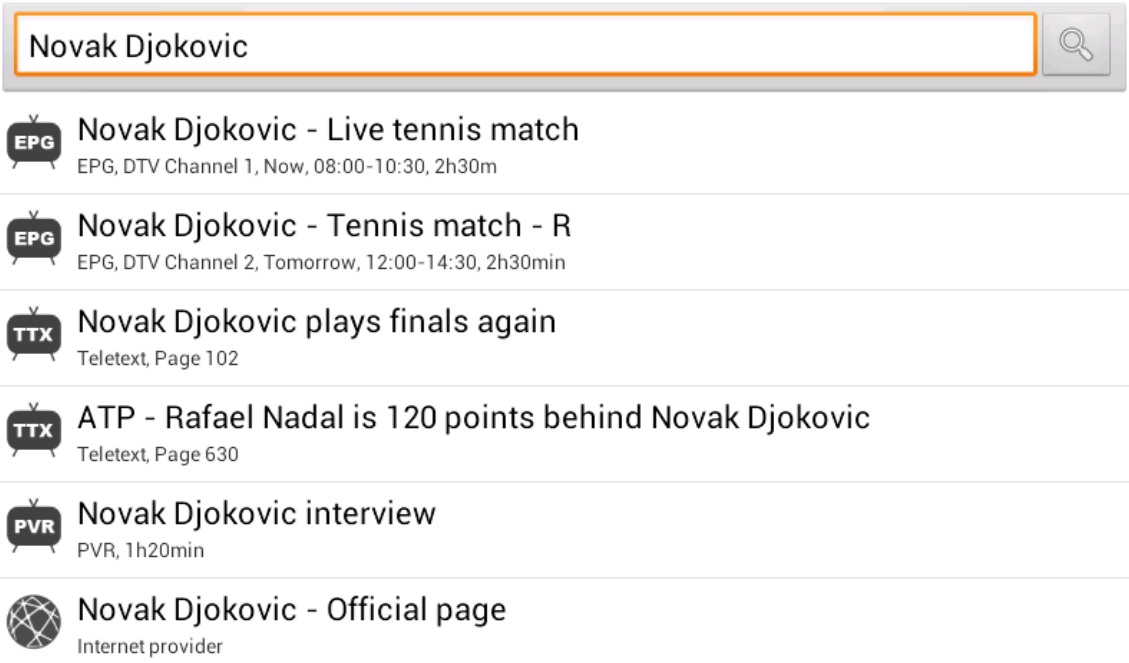
Pošto je kao osnova za razvoj TV softvera izabrano Android4TV okruženje, u cilju rešavanja prvog zadatka, ova aplikativna programska podrška će biti proširena odgovarajućim mehanizmima za pretragu TV sadržaja. Ovde će se definisati aplikativna programska sprega koja će pratiti principe Android4TV API i omogućiti nesmetano korišćenje obe sprege (osnovne i proširene).

U cilju rešavanja drugog zadatka, treba obezbediti podršku za dobavljanje i skladištenje pretraživih podataka u bazu podataka nad kojom će se vršiti pretraga. Ovo podrazumeva realizaciju servisa čije će zaduženje biti da održava ovakvu bazu podataka. Pretraga će se oslanjati na ovu bazu podataka a rezultati će biti dostupni korišćenjem proširene Android4TV API podrške.

Za potrebe trećeg zadatka biće napravljen dobavljač podataka za pretragu TV baziranih informacija iz signala digitalne televizije. Implementacija ovog dobavljača podataka će koristiti definisano proširenje za Android4TV API. Cilj je da na Android TV uređaju bude moguće pretraživati TV sadržaj na način na koji su korisnici Android uređaja već navikli. Tako će QSB pored prethodno navedenih primera pretrage sadržati i rezultate pretrage TV relevantnih informacija.

U okviru četvrtog zadatka treba napraviti minimalnu sistemsku podršku za reakciju sistema na odabrane rezultate pretrage. Ovo podrazumeva da se implementira Android aplikacija koja će reagovati na sistemske događaje izbora rezultata QSB aplikacije. Ova aplikacija će podržavati samo osnovne funkcionalnosti, neophodne da bi se eksperimentalno pokazalo da su rezultati pretrage validni.

Treba napomenuti da predloženo proširenje API programske podrške za pretragu nije ograničeno samo na QSB dobavljač podataka (iz trećeg zadatka). Ova aplikacija, u nekom smislu, predstavlja primer kako se predloženi API koristi. API se može iskoristiti i za proizvoljnu implementaciju pretrage u okviru neke Android aplikacije.



Slika 2.10 Prikaz QSB aplikacije sa implementiranim rešenjem za pretragu TV podataka

3. Pregled relevantnih informacija

Tržište potrošačke elektronike nudi mnoštvo raznovrsnih rešenja iz oblasti televizije u kojima i minimalna unapređenja mogu biti marketinški adut. S toga se proizvođači utrkuju u predstavljanju proizvoda kao inovativnog u svojoj kategoriji. Danas već postoje raznovrsna industrijska rešenja koja nude Android OS iskustvo na TV i STB uređajima

U ovom poglavlju ćemo videti koja su trenutna dostignuća razvoja oblasti digitalne televizije sa sistemima zasnovanim na Android OS, kao i u opšte TV sistemima koji podržavaju Java platformu. Takođe će biti obrazložene i teorijske osnove za pretraživanje podataka koji se prenose digitalnim TV transportnim tokom podataka.

3.1 Radovi na temu integracije Android OS platforme u TV uređaje

Iako je ova oblast relativno nova, postoje naučni radovi koji se bave problematikom integracije digitalnih TV servisa u Android operativni sistem. Sa stanovišta ovog rada, bitno je analizirati mogućnost integracije aplikativne podrške za pretragu TV sadržaja u predloženim rešenjima.

U [Lu] i [Kuzmanovic] prikazana su ovakva rešenja, pri čemu [Lu] predlaže sistem koji je specifičan za jednu fizičku arhitekturu. [Kuzmanovic] opisuje sistematičniji pristup, koji se bazira na sistemskoj izmeni grafičkog sistema, koji omogućuje prikazivanje DTV sadržaja na određenom delu ekrana TV prijemnika baziranom na Android OS. Ova rešenja ne nude standardizovanu podršku za razvoj TV aplikacija.

Sa druge strane, u [Vidakovic] i [Vidakovic2], prikazan je drugačiji pristup DTV integraciji u Android OS. Ovaj pristup ne zahteva izmene originalnog operativnog sistema i omogućuje rad na širokom skupu uređaja (ugrađeni uređaji, ali i personalni računari). Radovi

predlažu i Java API, ali i API za izvršni sloj koji se može koristiti kao osnova za standardizaciju DTV aplikacija pisanih Java programskim jezikom. Takođe, ovi radovi nastoje da postave standarde u načinu na koji se servisi digitalne televizije integrišu u ovaj operativni sistem. Rezultat istraživanja ovih radova je upravo objavljivanje javne Android4TV API programske podrške, koja će u ovom radu biti iskorišćena za integraciju TV funkcionalnosti.

3.2 Radovi na temu integracije TV bazirane pretrage u Android OS

Prethodno opisani radovi poslužili su kao inspiracija za dalji razvoj oblasti istraživanja razvoja TV aplikacija u Android okruženju. Tako, pored brojnih varijacija i unapređenja pomenutih radova, postoji i nekoliko koji se bave pretragom DTV sadržaja.

U svom radu, [Bjelic] se bavi implementacijom pretrage teletekst sadržaja na uređaju zasnovanom na Android OS, ili konkretnije na Google TV platformi. U pomenutom radu je takođe iskorišćena QSB aplikacija za inicijativu pretrage, samo u ovom slučaju to je Google TV QSB aplikacija, čiji je rad ograničena na Google TV platformu. Kada se uzme u obzir i činjenica da se ovde sama pretraga izvršava u DTV Middleware servisu (izvršni, platformski zavistan kod), dobijamo rešenje koje nije nezavisno i prenosivo na bilo koju Android platformu, već je vezano isključivo za Google TV platformu. Rešenje opisano u ovom radu takođe nudi i markiranje traženog upitnog teksta u samom prikazu teletekst stranice, standardnim mehanizmom bojenja pozadinske boje karaktera u teletekst grafičkom prikazu.

[Djukic] u svom radu predstavlja rešenje za implementaciju EPG pretrage na uređaju zasnovanom na Android OS. Ovde se takođe koristi QSB za predstavljanje rezultata i upit pretrage. Ovaj rad se oslanja na predložena rešenja iz [Vidakovic2] koja su proširena radi implementacije DTV pretrage. Kako se ovaj rad oslanja na predefinisanu programsku podrška za razvoj TV aplikacija (preteča Android4TV), sama programska podrška je proširena tako da su funkcije za pretragu dostupne aplikativnom domenu. Sam upit pretrage se (i ovde) propagira do sistemskog servisa DTV Middleware koji rezultate pretrage vraća putem Android Binder mehanizma. Kako je prenos podataka putem Android Binder mehanizma memorijski ograničen to ovde može predstavljati problem prilikom formiranja obimnijih lista rezultata.

U [Lukic2][Lukic] je predloženo rešenje iz [Vidakovic2] dodatno prošireno sa podrškom za pretragu DTV baziranih podataka. I ovde se koristi QSB aplikacija kod pretrage TV podataka. Principi implementacije i integracije programskog rešenja opisanog u ovom radu rešenje iz [Djukic] s tim što se pretraga EPG podataka proširuje na teletekst i PVR. Ovo dovodi do dodatnog proširenja aplikativne programske podrške i integracije dodatnih programskih sprega. Pretraživanje podataka u ovom radu je takođe implementirano u sistemskom servisu DTV

Middleware (kao i u dva prethodno opisana) s tim što je algoritam pretrage znatno efikasniji pošto je optimizovan primenom tehnike *edit* odstojanja između dva stringa [Abe].

Ovaj rad se nadovezuje, i u izvesnoj meri objedinjuje, opisane radove iz ove oblasti istraživanja. Predloženo rešenje je platformski nezavisno i koriste se samo javno dostupna okruženja za aplikativni razvoj.

Predloženo rešenje za implementaciju TV pretrage će se oslanjati na Android4TV implementaciju, kao kompletno rešenje za TV podršku na uređaju zasnovanom na Android platformi, i predstavljaće spregu između Android4TV aplikacije i mehanizma za pretragu u Android operativnom sistemu. Takođe, za potrebe ove implementacije će se proširiti i Android4TV API, kako bi se dodale funkcije za pretragu na nivou aplikativne podrške. Time je moguće da se pretraga vrši iz bilo koje aplikacije. Za potrebe demonstracije pretrage, neophodno je bilo i implementirati aplikaciju koja proširuje Android mehanizam pretrage a implementira Android4TV API (sa predloženim proširenjem). Na kraju, kako bi se rezultati pretrage predstavili korisniku, potrebno je još dodati i obezbediti aplikaciju koja će reagovati na rezultate pretrage, a u ovom slučaju to će biti proširena Android4TV aplikacija.

3.3 Ostala rešenja integracije Java bazirane platforme u TV sisteme

Iako je Android OS trenutno najpopularniji i najrasprostranjeniji operativni sistem baziran na Java platformi za programsku podršku, postoje i druga rešenja integracije Java programske podrške u TV uređaje.

3.3.1 Java TV

Java TV je Java bazirano softversko okruženje, dizajnirano za razvoj aplikacija u TV i STB uređajima. Trenutno se koristi za razvoj aplikacija za iTV platformu i podržava samo Java ME platformu. Java TV API je prilično siromašan, sa stanovišta podrške pune TV funkcionalnosti, i pruža samo osnovne funkcije za manipulaciju TV servisima.

U [Lucas] i [Brandao] je predloženo korišćenje [Java TV] standarda kao radnog okruženja prilikom rada sa DTV sadržajem na ugrađenim platformama. Ovaj pristup rešava problem platformski zavisne programske sprege pošto se koristi Java programski jezik za razvoj, dok se rad u realnom vremenu postiže upotrebom posebnog sloja izvršnih biblioteka koji je isključivo zadužen za parsiranje DTV podataka. Izvršni sloj je dostupan Java programskoj podrški kroz upotrebu JNI mehanizma. U ova dva rada je, takođe, predložen i skup različitih DTV aplikacija koje obezbeđuju različite servise kao što su mrežna povezanost, ali nedostaje standardizovano rešenje koje je primenljivo na različite uređaje, različitih proizvođača. Glavna mana upotrebe

Java TV standarda leži i u činjenici da je on deo Java ME (engl. Micro Edition) razvojne platforme koja je samo pod skup Java SE (engl. Standard Edition) platforme.

3.3.2 Google TV

Google TV predstavlja platformu, razvijenu od strane kompanije Google, koja je bazirana na Android operativnom sistemu koji proširuje određenim TV funkcionalnostima. Za razliku od Android operativnog sistema, programski kod Google TV platforme nije otvorenog tipa.

Slika 3.1 Google TV platform na Sony televizoru

Iako Google TV uređaji poseduju dobru integraciju DTV podataka sa naprednim funkcijama Android operativnog sistema, kao što je pretraga, ovi podaci i dalje ne dolaze iz DVB transportnog toka, već od strane namenskih servera (koje obezbeđuje i za čiju ažurnost se brine kompanija Google) putem Interneta. Ovo je ograničenje arhitekture programske podrške i kao posledicu ima potencijalnu neažurnost podataka (podaci u transportnom toku se značajno češće osvežavaju u odnosu na podatke koji dolaze putem Interneta).

3.4 Podaci TV pretrage

Postoje tri osnovna elementa svake implementacije softverske TV podrške na odgovarajućoj platformi:

- Podsistem za video/audio reprodukciju.
- Podsistem za obradu ostalih TV podataka (DSH).
- Korisnička TV aplikacija sa grafičkim prikazom.

Za ovaj rad su relevantni podaci koje obrađuje i generiše DSH podsistem i ovi podaci predstavljaju osnovicu za bazu podataka nad kojom će se vršiti pretraga. DSH je odgovoran za obradu kompleksnih podataka koji do uređaja stižu kroz digitalni TV signal i ovi podaci su grupisani po informacijama koje prenose u određene strukture podataka. Prenos i obrada ovih

informacija je definisana različitim standardima (za različite regione sveta). U ovom radu aspekt će biti na Evropskom standardu za digitalnu televiziju (DVB) i informacijama koje su propisane ovim standardom. Neke od bitnijih struktura podataka su:

- PES paketi
- PSI tabele
- Pod natpisi
- Teletekst
- EPG

Pored tri osnovna elementa za TV integracije, u savremenim rešenjima se podrazumeva i postojanje aplikacije za snimanje trenutno emitovanog digitalnog video sadržaja (PVR), koji se kasnije može reprodukovati. Snimanje videa se obično beleži na nekoj stalnoj memoriji kao što su: prenosive memorije sa USB konekcijom, memorijske SD kartice ili čak u na mrežnim „oblak“ servisima.

Kao kriterijumi za rezultate pretrage koju pokriva implementacija opisana u ovom radu uzimaju se tri kategorije podataka, a to su:

- EPG informacije.
- Teletekst sadržaj.
- PVR lista snimaka.

U nastavku sledi detaljniji opis ovih struktura.

3.4.1 Definicija i struktura EPG podataka

EPG predstavlja tekstualne podatke koji se prenose TV signalom i prenose detaljne informacije o trenutnim i predstojećim emisijama sa svih TV kanala (televizijskih i radio). EPG je deo Evropskog standarda za prenos podataka digitalne televizije. Postoje dve grupe EPG podataka:

- Trenutni odnosno sledeći (engl. Present/following), koja prenosi podatke o emisiji koja se trenutno prikazuje kao i emisiji koja sledi iza trenutne. Ovi podaci se osvežavaju na svake 2 sekunde.
- Budući, koja prenosi podatke o svim emisijama u budućnosti počev od trenutno prikazivane pa do 64 dana u budućnosti. Ovi podaci se osvežavaju na svakih 30 sekundi.

EPG informacije prenose predefinisani skup polja za svaku emisiju. Skup se sastoji od sledećih polja:

- Naziv emisije
- Vreme početka emisije

- Vreme završetka emisije
- Žanr kojem emisija pripada
- Preporučljiva starost gledaoca
- Kratak opis emisije, tekst dužine 256 karaktera
- Opširniji opis emisije, tekst dužine 3984 karaktera

Uobičajeno je da se EPG informacije predstavljaju korisniku televizora u tabelarnom grafičkom prikazu, gde će recimo, po vertikali da budu poređani TV kanali a u nastavku svakog TV kanala horizontalno bi bile poređane emisije. Tipična interakcija korisnika sa EPG tabelom podrazumeva kretanje kroz tabelu, prikazivanje dodatnih informacija o budućim emisijama i postavljanje podsetnika ili obeležavanje emisija za buduće snimanje.

Kako su EPG tekstualni podaci kojima korisnik često pristupa, praktično bi bilo omogućiti pretragu istih. U ovom radu se kao kriterijum za pretragu EPG podataka koristi kombinacija kratkog i opširnog opisa emisije.

3.4.2 Definicija i struktura teletekst podataka

Teletekst predstavlja tekstualne informacije koje se prenose PES paketima i podaci su organizovani u numerisane stranice. Teletekst stranicama se pristupa tako što korisnik unese trocifreni broj na daljinskom upravljaču nakon čega se stranica grafički prikaže. Stranica se sastoji od 23 tekstualne linije a svaka linija od po 40 karaktera.

Svaki teletekst PES paket je veličine 45 bajta sto omogućava da se svaka teletekst linija prenese jednim PES paketom. Jedan bajt u PES paketu predstavlja jedan tekstualni karakter dok preostalih 5 bajtova služe za određene kontrolne informacije, kao što su jezik ili set karaktera.

Teletekst dozvoljava da se emituje do 8 magazina, koji se identifikuju prvom cifrom trocifrenog broja stranice. Svaki magazin može sadržati do 256 stranica istovremeno, stranice u magazinu su numerisane dvocifrenim heksadecimalnim brojem. Tako na primer magazin 1 sadrži stranice od 100 do 1FF. U praksi, međutim, stranice identifikovane heksadecimalnim brojevima koji sadrže cifre A-F se ne koriste jer korisniku ove cifre u glavnom nisu dostupne na daljinskom upravljaču.

U ovom radu se kao kriterijum za pretragu teletekst podataka uzimaju informacija iz pojedinačnih stranica koje se mogu protumačiti kao tekst pogodan za čitanje. Na samom prikazu teletekst stranice se mogu takođe naći i karakteri koji pomažu pri izradi grafičke kompozicije ali u stvari nisu deo čitljivog teksta (razne strelice, ikonice, linije i ostali grafički elementi).

3.4.3 Definicija i struktura PVR podataka

Savremeni TV uređaji obično imaju mogućnost snimanja TV emisija u internu ili neku eksternu stalnu memoriju. Snimanje ovih podataka može biti manuelno, kada korisnik tokom gledala emisije pritisne dugme za snimanje; ili automatizovano, kada korisnik u EPG tabeli označi određene emisije u budućnosti za snimanje.

Pored snimanja audio i video sadržaja beleže se i EPG informacije snimaka. Ove informacije su, kao i u slučaju pretrage EPG sadržaja, iskorišćene kao kriterijum za pretragu snimljenog sadržaja. Ovo znači da će zahtev za pretragu PVR podataka kao povratnu vrednost imati spisak svih snimaka koji u sebi sadrže EPG događaj čiji opis odgovara traženom kriterijumu.

4. Pregled mera za ocenu kvaliteta rešenja

Fokus istraživanja ovog rada je definisanje aplikativnog okruženja u kome je moguće pretraživati TV podatke koji se prenose transportnim tokom podataka. Cilj je napraviti ovakvo okruženje i implementirati samu pretragu na Android OS platformi.

Prvo, uzimajući u obzir da je okruženje za aplikativni razvoj u Android OS bazirano na Java programskom jeziku, predloženo programsko rešenje će biti upoređeno sa ostalim aplikativnim programskim spregama za razvoj aplikacija u ovom okruženju. Ovakva analiza poređenja predloženog rešenja sa postojećim rešenjima biće prva mera za ocenu kvaliteta predloženog rešenja i naziva se mera **kompletnosti**. Mera kompletnosti će biti posmatrana kao skup podržanih funkcija (mogućnosti) merene aplikativne programske sprege, relevantnih za implementaciju pretrage TV sadržaja.

Dalje, uzimajući u obzir da na polju potrošačke elektronike u modernim TV/STB sistemima postoje slična rešenja, predloženo rešenje će biti analizirano sa aspekta **ažurnosti** i **performansi**. Ovakva analiza će dati uvid u prednosti i mane predloženog rešenja, naspram onih koji se, u trenutku pisanja ovog rada, mogu naći na tržištu. Mera ažurnosti treba da pokaže koliko su podaci, uslovno rečeno, tačniji, tj. sa kolikom kašnjenjem dostupni korisniku u odnosu na trenutak njihovog objavljivanja od strane emitera. Mera performansi treba da pokaže koje rešenje brže servira rezultate pretrage korisniku od trenutka kada je korisnik inicirao pretragu, nezavisno od toga koliko su podaci tačni. Ključna razlika između mera ažurnosti i performansi je što prva predstavlja kvalitativnu ocenu validnosti podataka dok druga predstavlja kvantitativni ocenu količine podataka.

4.1 Mere kompletnosti

Savremeni TV uređaji pružaju korisniku mnoge dodatne sadržaje pored klasičnog gledanja TV programa. Kao što je već rečeno u ranijim poglavljima, sve je češći trend da se ovakav TV

uređaj baira na Android OS. Činjenica je da standardna podrška za razvoj Android aplikacija ne pokriva mehanizme vezane za TV funkcionalnost. U ovom radu su već opisana neka od rešenja koja nude aplikativnu programsku spregu za integraciju TV funkcionalnosti u Android bazirane TV uređaju.

Kako je cilj ovog rada da predstavi rešenje za pretragu TV baziranih podataka na uređaju zasnovanom na Android OS, prevashodno je neophodno definisati funkcionalnosti koje trebaju biti obezbeđene za implementaciju ovakve aplikacije. Ovo se odnosi na neophodne metode za iniciranje pretrage EPG, teletekst i PVR sadržaja, kao i na funkcije za adekvatno prezentovanje rezultata pretrage. Tabela [Tabela 4.1] prikazuje spisak neophodnih funkcionalnosti koje su preduslov za izradu aplikacija za pretragu i prikazivanje rezultata pretrage.

Funkcionalnost	Razlog neophodnosti funkcionalnost
Reprodukcija MPEG video sadržaja	Ovo je preduslov za prikazivanje odabranih PVR rezultata.
Reprodukcija DTV sadržaja	Ovo je preduslov za prikazivanje sadašnjih EPG rezultata.
Pristup listi kanala	Ovo je preduslov za prikupljanje EPG i PVR podataka.
Pristup EPG sadržaju	Ovo je preduslov za prikupljanje EPG podataka.
Pretraga EPG sadržaja	Ovo je preduslov za pretragu EPG podataka.
Postavljanje EPG alarma	Ovo je preduslov za prikazivanje budućih EPG rezultata (postavljanje alarma za te događaje).
Prikazivanje teletekst sadržaja	Ovo je preduslov za prikazivanje teletekst rezultata.
Pristup teletekst podacima	Ovo je preduslov za prikupljanje teletekst podataka.
Pretraga teletekst podataka	Ovo je preduslov za pretragu teletekst podataka.
Snimanje DTV sadržaja (PVR)	Ovo je preduslov za postojanje PVR podataka.
Reprodukcija PVR sadržaja	Ovo je preduslov za prikazivanje PVR rezultata.
Pristup PVR lisi snimaka	Ovo je preduslov za prikupljanje PVR podataka.
Pretraga PVR sadržaja	Ovo je preduslov za pretragu PVR podataka.

Tabela 4.1 Funkcionalnosti neophodne za implementaciju aplikacije za pretragu

Cilj ove mere kvaliteta je da uporedi postojeće aplikativne programske sprege sa onom koja je definisana u ovom radu, radi analize mogućnosti izrade sistema za pretragu TV baziranih podataka.

4.2 Mere ažurnosti

Postojeća industrijska rešenja, koja nude pretraživanje EPG sadržaja, oslanjaju se na predodređene internet servise za dobavljanje EPG informacija, tako da ažurnost ovih informacija zavisi od ažurnosti baza podataka samih servera. Pod pretpostavkom da je najrelevantniji izvor EPG informacija onaj koji se prenosi DVB transportnim tokom, možemo reći da ovi internet servisi ne sadrže najnovije informacije u svakom trenutku niti najsvežije podatke u eventualnim promenama EPG rasporeda.

Pošto se implementacija EPG pretrage predstavljena u ovom radu oslanja upravo na DVB transportni tok kao jedini izvor EPG informacija, možemo pretpostaviti da će rezultati dobijeni ovom pretragom biti ažurniji u odnosu na one koje treba prvo da odobri i ažurira server u oblaku predodređenog internet servisa.

Da bi ispitali ovu pretpostavku, treba uporediti rezultate pretrage nekog od komercijalnih rešenja koji se oslanjaju na predodređene internet servise i rezultate dobijene pretragom informacija preuzetih direktno sa DVB transportnog toka podataka. Pretraga se treba izvršiti nad istim skupom podataka i pod uslovom da je neposredno pred iniciranje pretrage došlo do promene u EPG tabeli.

Period ažuriranja EPG podataka u DVB transportnom toku je definisan samim DVB standardom. EPG podaci se u DVB standardu definišu kroz strukturu podataka EIT tabela i podeljeni su u EIT pod-tabele. Periodi ažuriranja EIT pod-tabela su prikazani tabeli [Tabela 4.2].

EIT pod-tabela	Period ponavljanja (sekunde)
EIT trenutni/sledeći (aktuelni)	2
EIT trenutni/sledeći (ostali)	20
EIT budući (aktuelni događaji za prvi dan)	10
EIT budući (ostali događaji za prvi dan)	60
EIT budući (aktuelni događaji za naredne dane)	30
EIT budući (ostali događaji za naredne dane)	300
	Vrednosti definisane DVB standardom iz [ETSI]

Tabela 4.2 Periodi ažuriranja EIT pod-tabela

4.3 Mere performansi

I ova mera ocene kvaliteta (kao i prethodna, mera ažurnosti) treba da uporedi postojeća industrijska rešenja koja EPG informacije dobavljaju putem interneta i predloženog rešenja koje

EPG informacija dobavlja direktno iz DVB transportnog toka podataka. Ova analiza treba da pokaže da li se, i koliko, razlikuje brzina prikazivanja samih rezultata pretrage, od trenutka kada je zahtev za pretragom poslat sistemu do trenutka kada su rezultati obrađeni i prezentovani krajnjem korisniku.

Za razliku od mere ažurnosti, kod analize ponuđenih rešenja ovom metodom polazimo od pretpostavke da su podaci ažurni u oba slučaja (predodređeni internet servis raspolaže istim podacima kao i DVB transportni tok podataka) i pretraga pronalazi isti skup EPG događaja. Kada su ovi uslovi ispunjeni možemo imati relevantne podatke merenja.

Ova mera je važna jer odražava kvalitet ponuđenog rešenja sa stanovišta korisničkog iskustva.

5. Predloženo rešenje

Kao što je već pomenuto, cilj implementacije je obezbediti sistemskoj Android aplikaciji za globalnu pretragu, QSB, da prilikom zahteva za pretragu, pored ostalog sadržaja, prikaže i rezultate vezane za TV pretragu, a to su: EPG, teletekst i PVR podaci. U ovom poglavlju će biti opisana rešenja svih (u prethodnim poglavljima navedenih) zadataka koji stoje na putu ka ovom cilju:

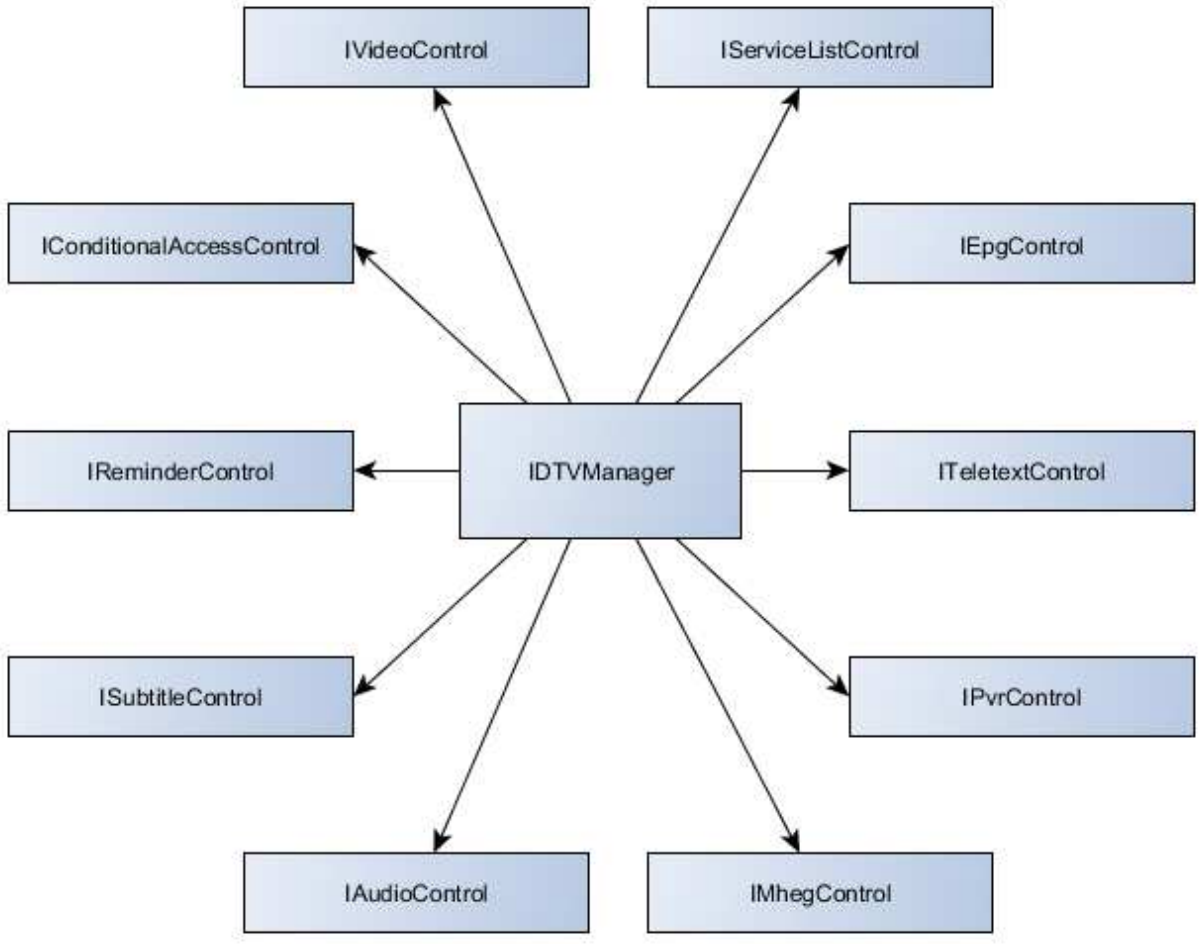
- Prvo će biti opisano proširenje Android4TV API sloja kome će biti dodata funkcionalnost za pretragu TV sadržaja.
- Zatim će biti opisana implementacija same funkcionalnosti Android4TV API-ja koja izvršava pretragu i kreira rezultate pretrage.
- Da bi omogućili QSB aplikaciji da pretraži ove podatke, implementirana je DTV Search Provider aplikacija koja kroz mehanizam za dobavljanje podataka preuzima zahtev pretrage i uspostavljanjem veze sa Android4TV API slojem, preuzima rezultate koje na kraju vraća kao formiranu listu u QSB aplikaciju.
- Na kraju treba napraviti Android aplikaciju koja će reagovati na rezultate TV pretrage i iste predstavi krajnjem korisniku na odgovarajući način. Ovo podrazumeva implementaciju standarde Android aplikacije koja će biti registrovana na određene šeme događaja a koristiće standardni (ne prošireni) Android4TV API za reprodukciju odgovarajućeg sadržaja.

5.1 Proširenje Android4TV API sloja

Kao što je već pomenuto, ovaj rad se oslanja na Android4TV razvojno okruženje da bi postigao upravljanje TV funkcijama Android baziranog uređaja. Android4TV softversko rešenje se sastoji od više slojeva implementacije (što je pomenuto u prethodnim poglavljima) a u ovom poglavlju je fokus na Android4TV API sloju.

Cilj prvog zadatka je proširiti Android4TV API tako da je moguće inicirati pretragu TV baziranog sadržaja. Ovo treba uraditi tako da osnovna funkcionalnost Android4TV API ostane netaknuta. Da bi se ovo postiglo, biće implementirana nova Java statička biblioteka – *android4tv-support-search.jar*. Pošto ova biblioteka predstavlja samo proširenje, aplikacija koja se bavi TV pretragom će morati da uključi osnovnu Android4TV API biblioteku a zatim i ovo njeno proširenje – **Android4TV Search API**.

Organizacija Android4TV API je podeljena u tzv. kontrolne sprežne klase. Svaki deo TV sistema koji predstavlja određenu funkcionalnu celinu se izdvaja kao jedna programska sprega (engl. Interface), i po pravilu to će biti Java sprežna klasa čije se ime završava za rečju „Control“. Na dijagramu [Slika 5.1] predstavljene su neke od komponenti klasa koje su od značaja za ovaj rad (ovo je samo pod-skup celog dijagrama sa).



Slika 5.1 Neke od osnovnih komponenti Android4TV API-ja

Kao što se iz dijagrama može videti, IDTVManager sprežna klasa predstavlja koren programske strukture i preko ove sprežne klase se dolazi do svih ostalih kontrolnih klasa. Implementacija ove sprežne klase je javna klasa DTVManager, i ovo je jedina sprežna klasa koja

se neposredno instancira javnim konstruktorom, sve ostale se instanciraju odgovarajućim metodama IDTVManager sprežne klase.

```
IDTVManager dtvManager = DTVManager.getInstance();
```

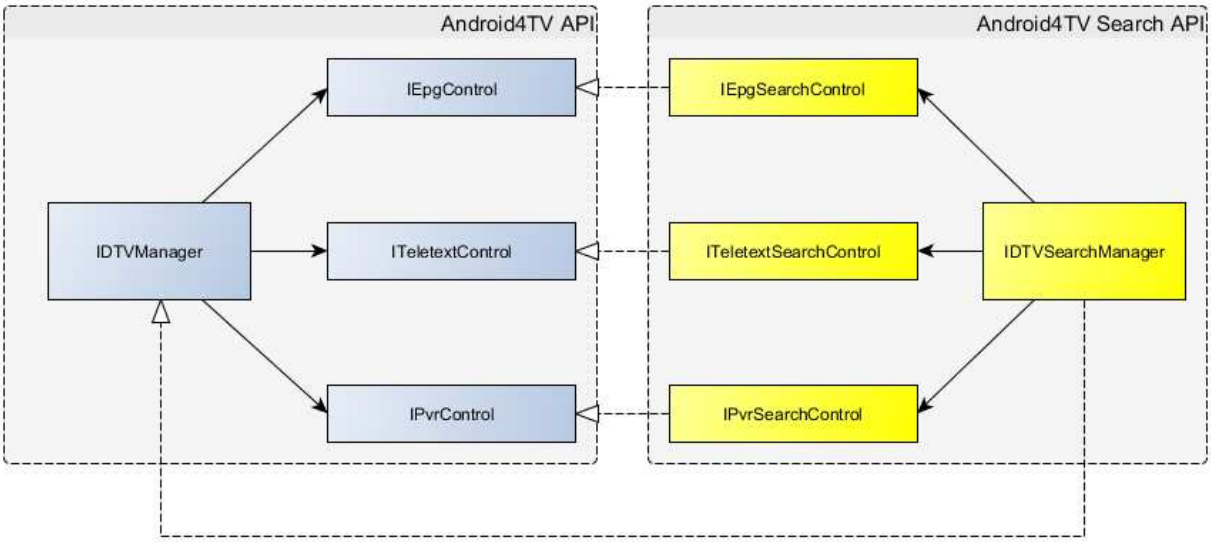
Kada je dostupna instanca klase IDTVManager, metodama ove klase se može pristupiti ostalim kontrolnim klasama. Kontrolne klase prikazane na prethodnom dijagramu obezbeđuju funkcionalnost TV uređaja koje su potrebne za implementaciju rešenja prikazanog u ovom radu. Slede primeri korišćenja ovih klasa:

- IServiceControl klasa – se koristi za pristup liste kanala dostupnih na TV uređaju i preko ove klase se vrši „prebacivanje“ (eng. Zapping) trenutnog kanala, tj prikazivanje željenog kanala.
- IEpgControl klasa – se koristi za pristup EPG tabeli koja je trenutno dostupna i koju je TV MW uspešno dobavio iz DTV transportnog toka podataka.
- ITeletextControl klasa – se koristi za prikazivanje teletekst stranica (ukoliko su dostupne) trenutnog kanala. Ovom klasom je takođe moguće i pristupiti samim podacima teleteksta (tekstualni podaci) ali ograničenje ovog modula je da se podaci teleteksta mogu videti samo za trenutno gledani kanal.
- IPvrControl klasa – se koristi za snimanje i puštanje snimljenog sadržaja. Ovom klasom je moguće doći do liste snimljenih emisija.
- IReminderControl klasa – se koristi za postavljanje alarma za željene emisije, koji će aplikacija istretirati. Ova klasa može pored fiksiranih vremenskih vrednosti da obeleži i alarm vezan za neki konkretan EPG događaj, tako da ako se promeni vreme početka označenog EPG događaja to će biti automatski propraćeno samim TV softverom.
- IConditionalAccessControl klasa – se koristi za kontrolu kodiranih kanala (engl. Scrambled) i ostalog zaštićenog sadržaja.
- IVideoControl klasa – se koristi za kontrolu video prikaza i podešavanja parametara za kvalitet slike (osvetljenje, kontrast, boja, itd.).
- IAudioControl klasa – se koristi za kontrolu audio komponente i podešavanja parametara reprodukcije zvuka (jačina, balans, nečujno stanje, itd.).
- ISubtitleControl klasa – se koristi za kontrolu ugrađenih pod-natpisa (može biti tekstualni ili slika) i podešavanje željenog jezika za prevod.
- IMhegControl klasa – se koristi za kontrolu MHEG-5 modula.

Proširenje Android4TV API-ja se odnosi na četiri sprežne klase iz ovog razvojnog okruženja, a to su:

- IDTVManager
- IEpgManager
- ITeletextManager
- IPvrManager

Ovo proširenje je ilustrovano dijagramom [Slika 5.2].

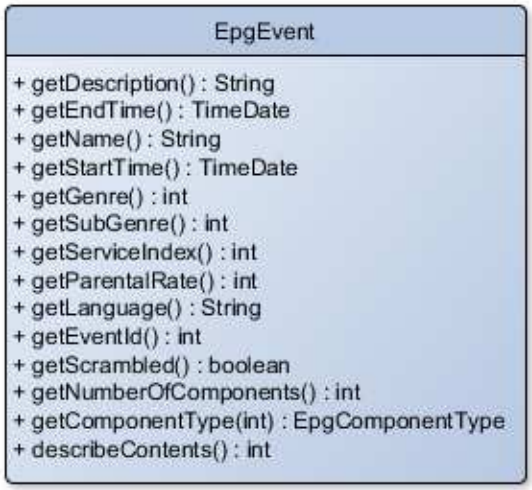


Slika 5.2 Proširenje Android4TV API kontrolnih klasa

Iz dijagrama se vidi da je IDTVManager sprežna klasa proširena (IDTVSearchManager) kako bi aplikacija za pretragu mogla pristupiti novim kontrolnim klasama, sa mogućnošću pretrage. Isto tako su proširene i tri navedene sprežne klase za kontrolu TV modula (IEpgControl, ITeletextControl i IPvrControl) novim sprežnim klasama koje dodaju metodu za dobavljanja liste rezultata pretrage na ovnovu zadatog upita.

5.1.1 Definicija IEpgSearchControl sprežne klase

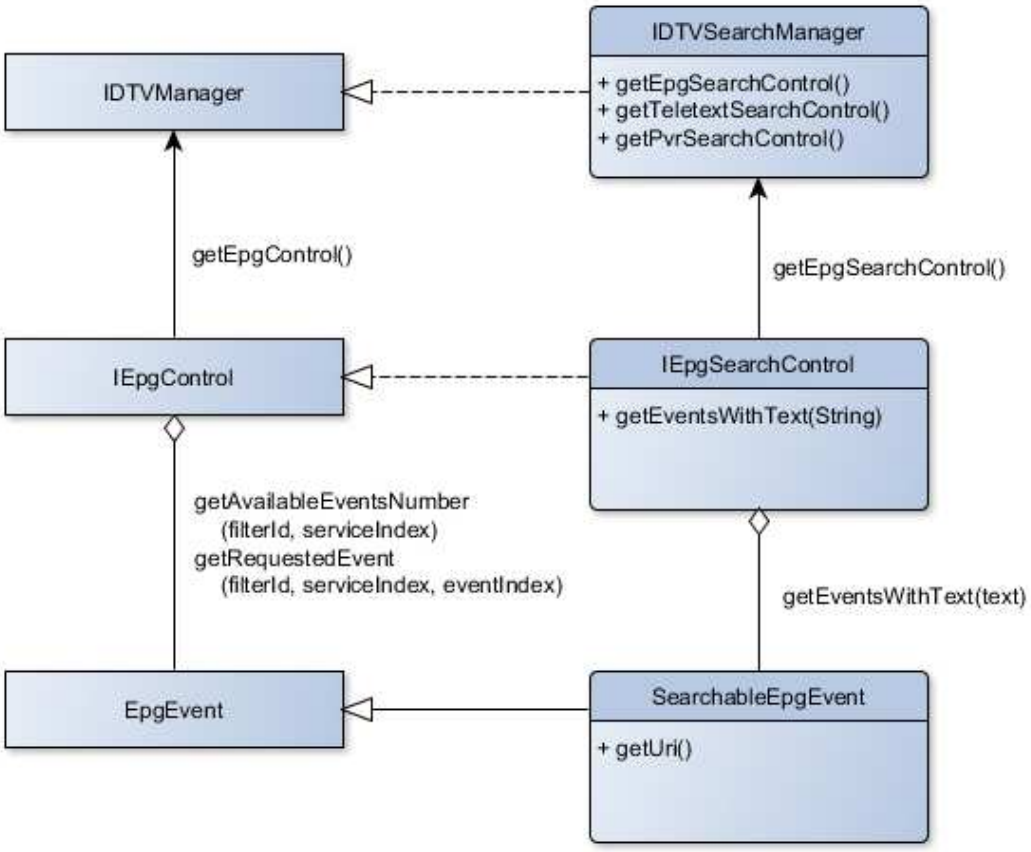
Android4TV API definiše posebnu spregu za pristup EPG podacima, a to je EPG kontrolna sprega – IEpgControl. Preko ove sprege može se pristupiti listi EPG događaja za zadate kanale, trenutnim informacijama o kanalu kao i informacijama o tome koja je trenutna i sledeća emisija. Svaki EPG događaj kojem se može pristupiti preko ove sprege definisan je klasom EpgEvent i ova klasa definiše funkcije za pristup svim informacijama o datom događaju, definisanim DVB standardom.



Slika 5.3 Definicija EpgEvent klase iz Android4TV API

Za potrebe pretrage EPG informacija uvedena je nova sprega koja predstavlja proširenje IEpgControl sprege – IEpgSearchControl. Ova nova sprega poseduje funkciju za iniciranje pretrage EPG sadržaja za zadati kriterijum pretrage a vraća listu pronađenih EPG događaja. Lista je formirana od objekata takođe nove klase, proširenja klase EpgEvent – SearchableEpgEvent.

UML klasni dijagram [Slika 5.4] prikazuje proširenja Android4TV API sloja za EPG pretragu.



Slika 5.4 EPG proširenja Android4TV API

Jedino proširenje koje donosi klasa SearchableEpgEvent u odnosu na osnovnu klasu EpgEvent je funkcija za dobavljanje jedinstvenog identifikatora resursa (engl. Uniform Resource Identifierza, URI) za traženi događaj. URI polje je direktno vezano za ponašanje Android sistema prilikom izbora datog rezultata pretrage. U URI polju EPG rezultata pretrage nalaze se informacije o sledećim parametrima EPG događaja:

- Identifikator kanala na kojem se emituje data emisija
- Vreme početka emisije
- Vreme završetka emisije
- Tekst kriterijuma pretrage

Jedan primer URI polja za upit pretrage za pojam „test“ bi mogao izgledati ovako:

```
tv://epg_search_result?service_id=101&epg_start_time=16:00:00&epg_end_time=17:15:00&query=test
```

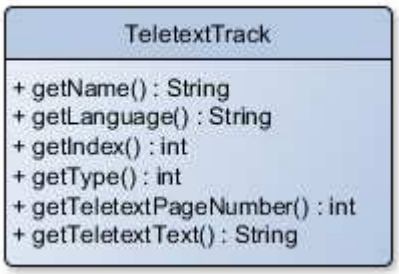
5.1.2 Definicija ITeletextSearchControl sprežne klase

Sa stanovišta TV aplikacije, kontrola teleteksta svodi se na tri jednostavne funkcije i u Android4TV API arhitekturi postoji posebna sprega za kontrolu teleteksta – ITeletextControl. Funkcije koje definiše ova sprega su:

- Zahtev za prikaz teletekst grafičkog prikaza za trenutni kanal.
- Prosleđivanje komandi sa daljinskog upravljača do teletekst modula.
- Zahtev za uklanjanje teleteksta grafičkog prikaza.

Ove funkcije se uvek odnose na kanal koji je trenutno aktivan i ukoliko pomenuti kanal ne podržava teletekst podatke funkcije će biti neuspešne. Ovo, uslovno rečeno, ograničenje se odražava i na implementaciju pretrage teleteksta i biće objašnjeno u nastavku poglavlja.

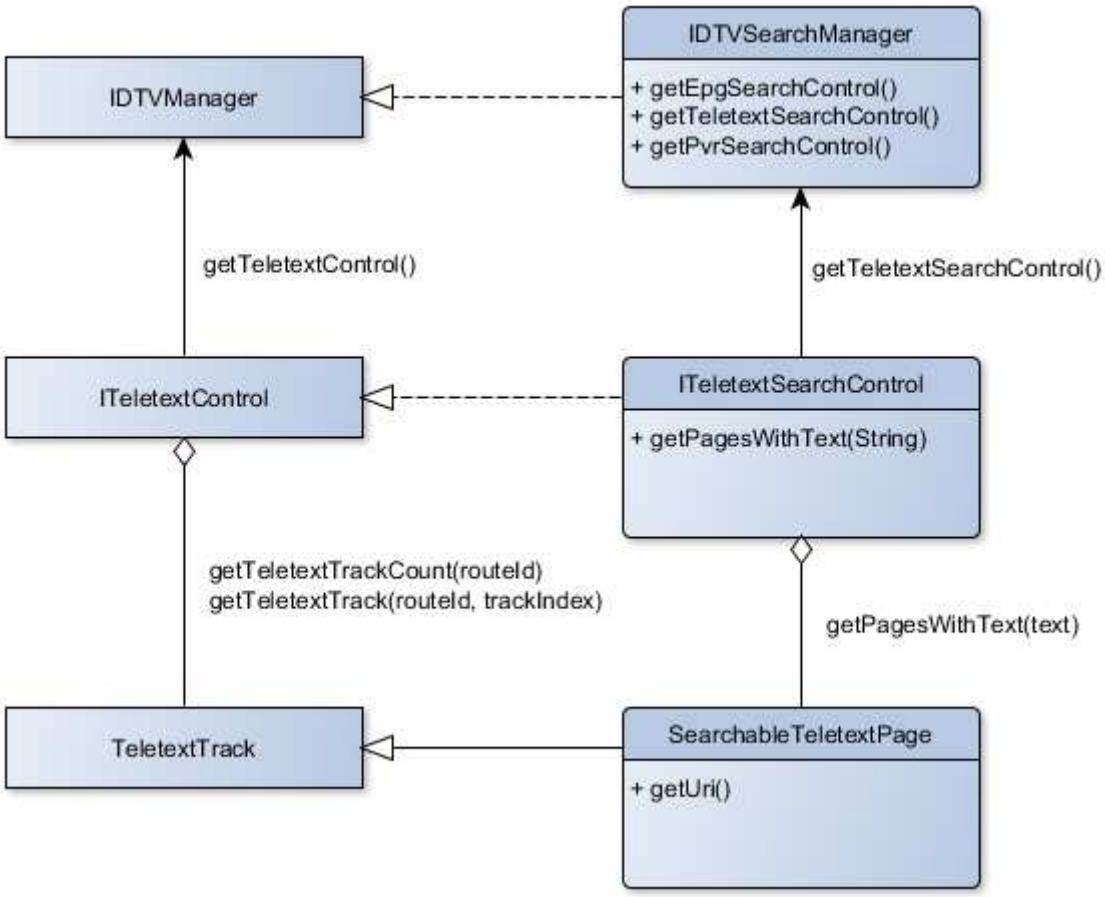
Teletekst stranica je u Android4TV API okruženju definisana klasom TeletextTrack. U ovoj klasi se čuvaju podaci o teletekst stranicama: redni broj stranice, naziv stranice i jezik stranice. Kada se dolazeći teletekst podaci iz DTV transportnog toka podataka obrade od strane DTV Middleware servisa, Android4TV API (putem ITeletextControl sprežne klase) raspolaže listom svih teletekst stranica koji su predstavljeni objektima klase TeletextTrack.



Slika 5.5 Definicija TeletextTrack klase iz Android4TV API

Za potrebe pretraživanja teletekst podataka uvedena je nova sprega kao proširenje sprege za kontrolu teleteksta koja uvodi funkcije za inicijativu pretrage teletekst podataka – ITeletextSearchControl. Slično kao kod sprege za pretragu EPG podataka, sprega za pretragu teletekst podataka implementira funkciju koja za zadati parametar kriterijuma pretrage vraća list pronađenih stranica. Ova lista je formirana od objekata klase SearchableTeletextPage koja je proširenje klase TeletextTrack.

UML klasni dijagram [Slika 5.6] prikazuje proširenja Android4TV API sloja za teletekst pretragu.



Slika 5.6 Teletekst proširenja Android4TV API

Kao i kod pretrage EPG podataka, za krajnji rezultat pretrage teleteksta sa stanovišta QSB aplikacije je neophodno postavljanje URI polja rezultata. Klasa SearchableTeletextPage definiše funkciju za dobavljanje URI polja za traženi događaj. U URI polju teletekst rezultata pretrage nalaze se informacije o sledećim parametrima teletekst stranice:

- Identifikator kanala na kojem se nalazi pronađena teletekst strana
- Broj teletekst strane na kome se nalazi pronađeni tekst
- Tekst kriterijuma pretrage

Jedan primer URI polja za upit pretrage za pojam „test“ bi mogao izgledati ovako:

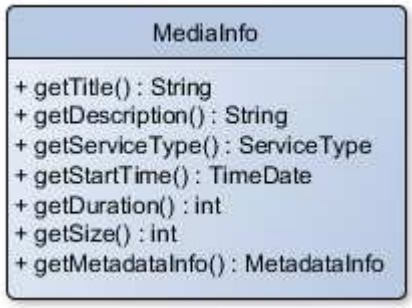
```
tv://teletext_search_result?service_id=101&teletext_page_number=200&query=test
```

5.1.3 Definicija IPvrSearchControl klase

PVR je kompleksan modul zadužen za snimanje i reprodukciju digitalnog audio i video sadržaja. Ovaj modul takođe pokriva i realizaciju naprednih scenarija, vezanih za snimanje/reprodukciju videa, kao što su:

- Zakasnela reprodukcija trenutno emitovanog video sadržaja (eng. Time-shifting).
- Pozadinsko snimanje video sadržaja koji trenutno nije aktivan.
- Neprimetnog snimanja video sadržaja dok je TV uređaj isključen (eng. Pseudo stand-by recording).

Sve navedene funkcije PVR modula dostupne su Android4TV API sloju kroz posebnu spregu za PVR kontrolu – IPvrControl. Od interesa za ovaj rad je samo deo PVR modula koji se bavi reprodukcijom snimljenog sadržaja, a to su funkcije za pregled spiskova snimljenog sadržaja i pokretanje reprodukcije pojedinih snimaka. Android4TV API definiše pomoćnu klasu koja jednoznačno određuje svaki pojedinačni snimak i povezuje ga sa datotekom snimka – MediaInfo.

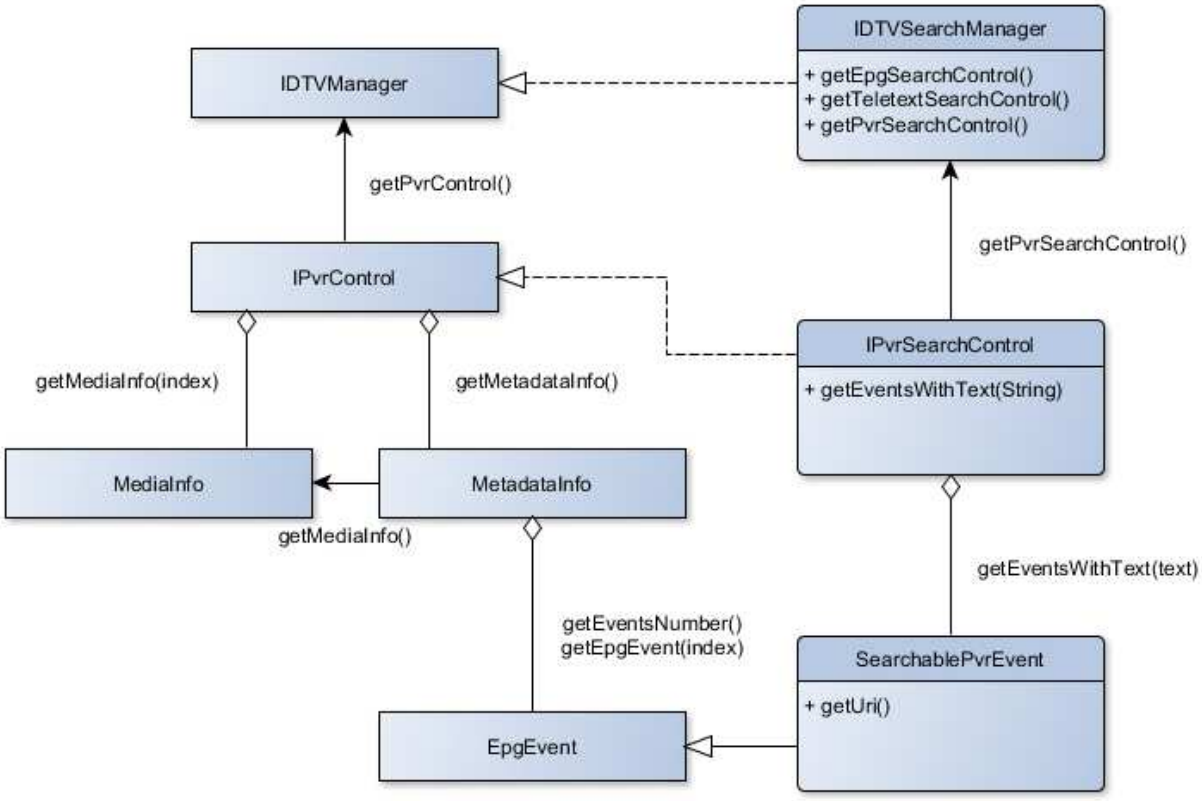


Slika 5.7 Definicija MediaInfo klase iz Android4TV API

Ovde treba napomenuti da dužina snimanja nije ograničena vremenski (već samo koliko memorijski kapacitet to dopušta) i može se desiti da u toku snimanja trenutna emisija se završi i počne nova. S toga svaka snimljena PVR datoteka može sadržati više EPG događaja, u zavisnosti od dužine snimka i promene emisija tokom snimanja. Android4TV API obezbeđuje listu EPG događaja (objekti klase EpgEvent) kojoj se može pristupiti za svaku snimljenu PVR datoteku. Snimljeni EPG događaji su deo MetadataInfo strukture (struktura koja čuva dodatne podatke vezane za snimljene fajlove) koja je dostupna za svaku instancu MediaInfo klase.

Za potrebe PVR pretrage uvedena je nova sprega koja predstavlja proširenje IPvrControl sprege – IPvrSearchControl. Ova nova sprega definiše funkciju za inicijativu pretrage PVR sadržaja za zadati kriterijum pretrage a vraća listu PVR snimljenih događaja. Lista je formirana od objekata takođe nove klase, proširenja klase EpgEvent – SearchablePvrEvent.

UML klasni dijagram [Slika 5.8] prikazuje proširenja Android4TV API sloja za PVR pretragu:



Slika 5.8 PVR proširenja Android4TV API

Kao i kod pretrage EPG i teletekst podataka, za krajnji rezultat PVR pretrage neophodno je generisanje URI polja rezultata kojem se može pristupiti preko funkcije za dobavljanje URI polja u klasi SearchablePvrEvent. U URI polju PVR rezultata pretrage nalaze se informacije o sledećim parametrima PVR datoteke:

- Identifikator PVR datoteke.
- Vreme početka pronađene emisije unutar PVR datoteke (izraženo u sekundama).
- Tekst kriterijuma pretrage.

Jedan primer URI polja za upit pretrage za pojam „test“ bi mogao izgledati ovako:

```
tv://pvr_search_result?pvr_file_id=12345&pvr_time_offset=60&query=test
```

5.2 Implementacija pretrage TV sadržaja

Pošto je u prethodnom poglavlju definisan API nivo koji se koristi za pretragu TV sadržaja, potrebno je implementirati taj API i zaista pretražiti TV sadržaj. Ova implementacija će se oslanjati na postojeće Android4TV mehanizme i koristiti Android4TV API sloj za dobavljanje TV podataka.

Podacima, koje ćemo pretraživati, su u Android4TV API okruženju dostupni isključivo kao rezultati poziva odgovarajućih funkcija. Tako na primer, da bi izlistali EPG događaje za odgovarajući TV kanal, pored identifikacionog broja kanala moramo znati i ukupan broj EPG događaja za taj kanal i zatim iterativno pozivali funkciju za dobavljanje EPG događaja. Isti princip važi i za ostale podatke (PVR i teletekst). Sledi primer listanja EPG događaja:

```
int eventsCount = epgControl.getAvailableEventsNumber(filterId, serviceId);
for (int eventIndex = 0; eventIndex < eventsCount; eventIndex++) {
       EpgEvent event = epgControl.getRequestedEvent(filterId, serviceId, eventIndex);
}
```

Ovakav princip, pristupa podacima indeksiranjem, nije pogodan za implementaciju pretrage čiji je kriterijum tekst (String). Za implementaciju ovakve pretrage neophodno je koristiti strukturu podataka čiji je identifikacioni ključ dosta složeniji od samo jednom brojčanog indeksa. U ovakvoj situaciji, da bi maksimalno optimizovali vreme pretraživanja, koristiće se SQL baza podataka. Android operativni sistem obezbeđuje SQLite mehanizam za rukovanje bazama podataka, i isti je dostupan kroz standardno javno Android razvojno okruženje.

Za potrebe ove impementacije biće napravljen Java servis, **TV Search Service**, čiji je posao da kreira i održava QSLite bazu podataka nad kojom će se vršiti pretraga. Tako će se funkcije za pretragu svoditi na standardno SQL upite nad bazom koju ovaj servis održava.

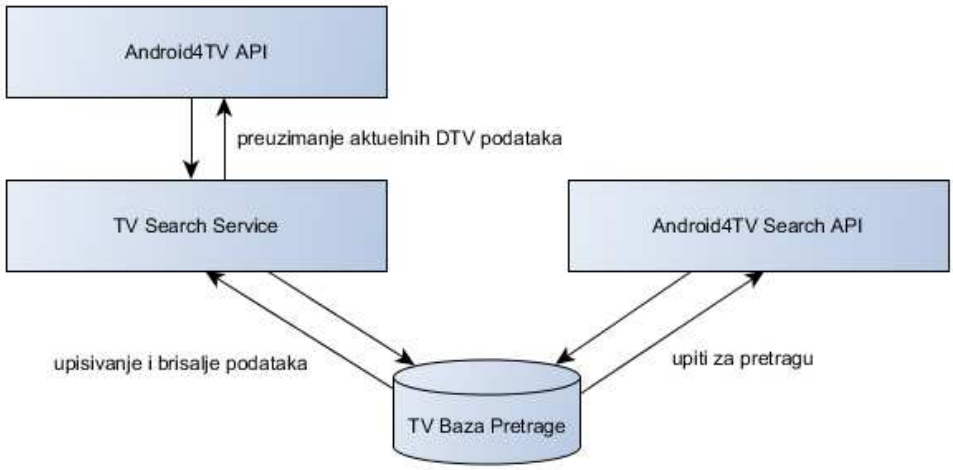
5.2.1 Servis za održavanje DTV baze podataka

Kao što je rečeno, zadatak ovog servisa je da održava TV bazu podataka. Ovo održavanje podrazumeva da se:

- Ako ne postoji, baza se kreira i popuni trenutno dostupnim podacima.
- Periodično se osvežavaju postojeći podaci.
- Periodično se dodaju novi podaci.
- Periodično se brišu oni podaci koji više ne treba da budu dostupni.

Nezavisno od ovog servisa, funkcije za pretragu će imati direktan pristup ovoj bazi i vršiti upite za pretragu. Na taj način je održavanje baze nezavisno od same pretrage. Na dijagramu [Slika 5.9] prikazana je međuzavisnost servisa i sprežnog sloja od baze podataka.

Baza se sastoji od tri tabele, po jedna za svaki tip podataka (EPG, teletekst i PVR). Polja ovih tabela odgovaraju podacima odgovarajućih klasa iz Android4TV API sloja. Svaka od ovih tabela se održava ponaosob a njihova struktura, kao i tehnike osvežavanja ovih podataka, će biti opisana u narednim poglavljima.



Slika 5.9 Organizaciona shema rešenja TV pretrage

5.2.1.1 Ažuriranje EPG tabele

EPG tabela se sastoji se od polja prikazanih u tabeli [Tabela 5.1].

Naziv kolone	Tip podatka	Opis
_id	INTEGER	Identifikator polja u SQLite bazi.
event_id	INTEGER	Identifikacioni broj događaja.
event_name	TEXT	Naziv događaja.
event_description	TEXT	Opis događaja, podatak koji se pretražuje.
event_start_time	INTEGER	Vreme početka događaja.
event_end_time	INTEGER	Vreme završetka događaja.
service_id	INTEGER	Identifikacioni broj servisa.

Tabela 5.1 Struktura EPG tabele

Polja EPG tabele su direktno povezana sa poljima EpgEvent klase Android4TV API-ja. Osvežavanje podrazumeva usklađivanje baze podataka sa podacima koji su trenutno dostupni kroz Android4TV API. Postoje tri mehanizma za osvežavanje EPG tabele:

- Kreiranje nove EPG tabele i unošenje inicijalnih podataka.
- Brisanje zastarelih događaja iz EPG tabele.
- Unošenje novih događaja u EPG tabelu.

Ukoliko EPG tabela ne postoji potrebno je kreirati novu tabelu u bazi podataka i popuniti je svežim informacijama. Unošenje inicijalnih podataka podrazumeva da se iterativno prelistaju svi dostupni kanali i za svaku unesu EPG događaji u EPG tabelu. Za inicijalno popunjavanje EPG tabele koristi se sledeća funkcija (oslanja se na funkciju za ažuriranje podataka koja će biti prikazana kasnije):

```
public void createEpgData() {
```

```
	int servicesCount = serviceControl.getServiceListCount(listIndex);
	for (int serviceIndex = 0; serviceIndex < servicesCount; serviceIndex++) {
		updateEpgData(serviceIndex);
	}
}
```

Android4TV API obezbeđuje mehanizam za detekciju promena u EIT tabeli DVB transportnog toka podataka, što znači da možemo znati kada je došlo do promene u tabeli aktuelnih EPG podataka. Za implementaciju ovog mehanizma koristi se povratna sprega IEpgCallback, koja se implementira u lokalnoj klasi a zatim se instanca ove povratne sprege registruje u IEpgControl kontrolnoj klasi:

```
epgControl.registerCallback(instanceOfIEpgCallback, filterId);
```

Povratna sprega IEpgCallback definiše metode kroz koje stižu obaveštenja o eventualnim promenama u aktuelnim EPG podacima. Metode koje su od interesa u ovoj implementaciji su:

- pfEventChanged – koja obaveštava da je do promene došlo u sadašnjem odnosno sledećem događaju.
- scEventChanged – koja obaveštava da je do promene došlo u budućem događaju.

Implementacija je zamišljena da ukoliko je došlo do promene u sadašnjem događaju, treba obrisati zastarele podatke, pošto to u glavnom znači da je trenutni događaj istekao tako da se može obrisati. Za ovakvu implementaciju je odlučeno zato što je povratna informacija o promeni za sadašnji događaj samo identifikacioni broj servisa, tako da se ne može znati koji se događaj promenio već samo na kom kanalu je došlo do promene. Sledi implementacija ove metode:

```
@Override
public void pfEventChanged(int filterID, int serviceIndex) {
	service.discardEpgData(serviceIndex);
}
```

Sama funkcija za brisanje zastarelih servisa treba samo da obriše sve događaje iz EPG tabele za zadati kanal, koji su u prošlosti:

```
public void discardEpgData(int serviceIndex) {
	long nowTime = setupControl.getTimeDate().getCalendar()
			.getTimeInMillis();
	getContentResolver().delete(
			EPG_TABLE_URL,
			EVENT_END_TIME + " < ? AND " + SERVICE_ID + " = ?",
			new String[] { String.valueOf(nowTime),
					String.valueOf(serviceIndex) });
}
```

Metoda povratne sprege za obaveštenje o promeni budućeg EPG događaja takođe za povratnu informaciju ima samo identifikacioni broj servisa na kome je došlo do događaja. Pošto je ovde nemoguće utvrditi koji je događaj promenjen, a da se ne uporede svi podaci iz baze sa aktuelnim EPG podacima, implementacija će biti uprošćena. U ovom slučaju potrebno je obrisati sve događaje EPG tabele za zadati kanal a potom ažurirati podatke, preuzevši sve aktuelne događaje zadatog kanala. Sledi implementacija ove metode:

```
@Override
public void scEventChanged(int filterID, int serviceIndex) {
	service.deleteEpgData(serviceIndex);
	service.updateEpgData(serviceIndex);
}
```

Sama funkcija za brisanje treba da obriše sve događaje iz EPG tabele za zadati kanal:

```
public void deleteEpgData(int serviceIndex) {
	getContentResolver().delete(EPG_TABLE_URL,
			SERVICE_ID + " = " + serviceIndex, null);
}
```

Dok funkcija za ažuriranje podataka treba iterativno da prođe kroz sve elemente niza EPG događaja vezanih za zadati kanal i doda relevantne podatke u bazu podataka. Ova funkcija se takođe koristi i kod inicijalnog popunjavanja podataka prilikom kreiranja EPG tabele. Sledi implementacija funkcije za ažuriranje EPG tabele za zadati kanal:

```
public void updateEpgData(int serviceIndex) {
	int eventsCount = epgControl.getAvailableEventsNumber(filterId,
			serviceIndex);
	ContentValues values[] = new ContentValues[eventsCount];
	for (int eventIndex = 0; eventIndex < eventsCount; eventIndex++) {
		EpgEvent event = epgControl.getRequestedEvent(filterId,
				serviceIndex, eventIndex);
		long startTime = event.getStartTime().getCalendar()
				.getTimeInMillis();
		long endTime = event.getEndTime().getCalendar().getTimeInMillis();
		ContentValues value = new ContentValues();
		value.put(EVENT_ID, event.getEventId());
		value.put(EVENT_NAME, event.getName());
		value.put(EVENT_DESCRIPTION, event.getDescription());
		value.put(EVENT_START_TIME, startTime);
		value.put(EVENT_END_TIME, endTime);
		value.put(SERVICE_ID, serviceIndex);
		values[eventIndex] = value;
	}
	getContentResolver().bulkInsert(EPG_TABLE_URL, values);
}
```

Ovde je prikazano jedno rešenje za implementaciju mehanizama ažuriranja EPG tabele. Zbog ograničenja samih Android4TV API metoda za prijavljivanje promena u EIT tabeli, implementacija nije najoptimalnije izvedena. Bolje rešenje bi bilo ažurirati samo podatke koji su se promenili a ne prilikom promene ažurirati sve podatke za zadati kanal. Kako je krajnji cilj ovog rada definisanje mehanizama za pretragu TV baziranih podataka, sama implementacija pretrage je implementirana na ovaj način a usavršavanje optimizacije ostaje otvoreno za neka buduća istraživanja.

5.2.1.2 Ažuriranje teletekst tabele

Teletekst tabela se sastoji se od polja prikazanih u tabeli [Tabela 5.2].

Naziv kolone	Tip podatka	Opis
_id	INTEGER	Identifikator polja u SQLite bazi.
page_index	INTEGER	Identifikacioni broj teletekst stranice.
page_text	TEXT	Tekst stranice, podatak koji se pretražuje.
page_number	INTEGER	Redni broj teletekst stranice.

Tabela 5.2 Struktura teletekst tabele

Polja teletekst tabele su direktno povezana sa poljima TeletextTrack klase u Android4TV API. Zbog ograničenja Android4TV API (a i DTV Middleware servisa) teletekst podaci su dostupni samo za TV servis koji je trenutno aktivan, tako da se ne beleži identifikacioni broj servisa na kojem se stranica nalazi. Ovo znači da je pretraga teleteksta moguća samo za teletekst podatke trenutno aktivnog kanala. Iz ovog razloga prilikom promene kanala potrebno je obrisati sve podatke iz teletekst tabele, a zatim ažurirati tabelu i popuniti je novim podacima.

Android4TV API obezbeđuje mehanizam za detekciju promena kanala, tako da će isti biti iskorišćen za adekvatno ažuriranja teletekst tabele. Za implementaciju ovog mehanizma koristi se povratna sprega IServiceCallback, koja se implementira u lokalnoj klasi a zatim se instanca ove povratne sprege registruje u IServiceControl kontrolnoj klasi:

```
serviceControl.registerCallback(instanceOfIServiceCalback);
```

Povratna sprega IServiceCallback definiše metode kroz koje stižu obaveštenja o promenama aktivnog kanala, promenama statusa aktivnog kanala kao i promena vezane za listu kanala. Metoda koja je od interesa u ovoj implementaciji je channelChangeStatus, i ona obaveštava da je trenutni kanal promenjen.

Zbog dinamičkog karaktera teletekst podataka, osvežavanje teletekst tabele se mora periodično obnavljati. Osvežavanje podrazumeva kompletno brisanje svog sadržaja i ponovno

ažuriranje podataka, preuzetih iz Android4TV API. Sledi implementacija metode povratne sprege za promenu kanala:

```
@Override
public void channelChangeStatus(int liveRoute, boolean channelChanged) {
	service.deleteTeletextData();
	scheduler.scheduleAtFixedRate(new Runnable() {

		@Override
		public void run() {
			service.deleteTeletextData();
			service.updateTeletextData();
		}
	}, 30, 5 * 60, TimeUnit.SECONDS);
}
```

Neposredno nakon promene kanala treba obrisati stare teletekst podatke, pošto su oni sada nevažeći. Zatim se zakazuje periodično izvršavanje funkcija za brisanje i ažuriranje teletekst tabele. Kako teletekst podaci nisu kompletni neposredno nakon promene kanala, već je potrebno izvesno vreme za obradu teletekst podataka pristiglih kroz DVB transportni tok, inicijalno izvršavanje ažuriranja je odloženo za 30 sekundi. Nakon toga, periodično se svakih 5 minuta osvežava kompletna teletekst tabela. Ove vremenske vrednosti su dobijene eksperimentalnim istraživanjem ponašanja implementiranog rešenja.

Funkcija za brisanje teletekst podataka svodi se na jednostavno brisanje svih polja teletekst tabele:

```
public void deleteTeletextData() {
	getContentResolver().delete(TELETEXT_TABLE_URL, null, null);
}
```

Funkcija za ažuriranje podataka treba iterativno da prođe kroz sve elemente niza teletekst stranica vezanih i doda relevantne podatke u bazu podataka. Sledi implementacija funkcije za ažuriranje teletekst tabele:

```
public void updateTeletextData() {
	int traksCount = ttxControl.getTeletextTrackCount(routeId);
	ContentValues values[] = new ContentValues[traksCount];
	for (int trackIndex = 0; trackIndex < traksCount; trackIndex++) {
		TeletextTrack track = ttxControl.getTeletextTrack(routeId,
				trackIndex);
		ContentValues value = new ContentValues();
		value.put(PAGE_INDEX, track.getIndex());
		value.put(PAGE_TEXT, track.getTeletextText());
		value.put(PAGE_NUMBER, track.getTeletextPageNumber());
		values[trackIndex] = value;
```

```
        }
        getContentResolver().bulkInsert(TELETEXT_TABLE_URL, values);
}
```

5.2.1.3 Ažuriranje PVR tabele

PVR tabela se sastoji se od polja prikazanih u tabeli [Tabela 5.3].

Naziv kolone	Tip podatka	Opis
_id	INTEGER	Identifikator polja u SQLite bazi.
pvr_event_file_id	INTEGER	Identifikator PVR fajla u kome se događaj nalazi.
pvr_event_index	INTEGER	Identifikacioni broj događaja.
pvr_event_description	TEXT	Opis događaja, podatak koji se pretražuje.
pvr_event_offset_time	INTEGER	Koliko je početak događaja udaljen od početka snimanja (u sekundama).

Tabela 5.3 Struktura PVR tabele

Polja PVR tabele odnose se na snimljeni EPG događaj koji pripada PVR fajlu snimku. Pošto se PVR datoteke snimaju na medijumu stalne memorije (u Android OS je to USB memorijski uređaj ili SD/MicroSD kartica) osvežavanje podrazumeva usklađivanje baze podataka sa podacima koji se nalaze na trenutnom memorijskom medijumu. Postoje tri mehanizma za osvežavanje PVR tabele:

- Kreiranje kompletne PVR tabele prilikom zamene memorijskog medijuma.
- Brisanje događaja za obrisani PVR fajl.
- Unošenje novih događaja za kreirani PVR fajl.

Kreiranje PVR tabele se izvršava u dva slučaja. Prvi je ukoliko PVR tabela ne postoji pa je potrebno inicijalno je kreirati, ovo se dešava na prvo pokretanje servisa za ažuriranje baze. Drugi slučaj je prilikom zamene memorijskog medijuma. Standardna Android aplikativna podrška obezbeđuje mehanizam za detekciju uključivanja, odnosno isključivanja, memorijskih medijuma u sistemu. Za detekciju ovih događaja koristi se mehanizam za primanje sistemskih poruka u Android OS, a sama akcija se implementira preko sprege za prijem emitovanih poruka (BroadcastReceiver). Sledi implementacija za registrovanja ovog događaja:

```
IntentFilter filter = new IntentFilter();
filter.addAction(Intent.ACTION_MEDIA_MOUNTED);
filter.addAction(Intent.ACTION_MEDIA_UNMOUNTED);
filter.addDataScheme("file");
filter.setPriority(PRIORITY);
registerReceiver(mediaBroadcastReceiver, filter);
```

Kada je događaj za detekciju promena memorijskih medijuma registrovan, eventualne promene u sistemu će pozvati metodu registrovane programske sprege. Ova metoda treba da obriše sve PVR događaje i kreira preuzme postojeće primenom Android4TV API funkcija za dobavljanje informacija o PVR događajima. Sledi implementacija ove metode:

```
@Override
public void onReceive(Context context, Intent intent) {
	service.clearPvrData();
	service.createPvrData();
}
```

Čišćenje PVR podataka podrazumeva brisanje svih podataka iz PVR tabele. Kod kreiranja PVR podataka potrebno je iterativno proći kroz sve zabeležene PVR fajlove i ažurirati podatke svakog od njih. Sledi implementacija funkcija za čišćenje i kreiranje PVR podataka (ažuriranje PVR fajla će biti prikazano naknadno):

```
public void clearPvrData() {
	getContentResolver().delete(PVR_TABLE_URL, null, null);
}

public void createPvrData() {
	int mediaCount = pvrControl.updateMediaList();
	for (int mediaIndex = 0; mediaIndex < mediaCount; mediaIndex++) {
		updatePvrData(mediaIndex);
	}
}
```

Android4TV API poseduje mehanizam za detekciju brisanja (od strane korisnika) i kreiranja novih (nakon završenog snimanja) PVR medijskih fajlova. Ovaj mehanizam je iskorišćen za optimalno ažuriranje PVR tabele. Za implementaciju ovog mehanizma koristi se povratna sprega IPvrCallback, koja se implementira u lokalnoj klasi a zatim se instanca ove povratne sprege registruje u IPvrControl kontrolnoj klasi:

```
pvrControl.registerCallback(instanceOfIPvrCallback);
```

Povratna sprega IPvrCallback definiše brojne metode kroz koje stižu obaveštenja o promeni različitih stanja u PVR modulu. Metode od interesa u ovoj implementaciji su:

- eventMediaAdd – koja obaveštava da je dodat novi PVR medijski fajl;
- eventMediaRemove – koja obaveštava da je PVR medijski fajl obrisan.

Android4TV API definiše povratne vrednosti za obe metode koje obaveštavaju o naslovu referentnog PVR fajla. Pošto se ova implementacija oslanja na identifikacioni broj fajla, pre preduzimanja akcije neophodno je pronaći ovaj identifikacioni broj (na osnovu naziva fajla) a zatim ažurirati tabelu. U pozivu metode za brisanje fajla treba obrisati sve PVR događaje vezane

za ovaj fajl, dok u metodi za dodavanje novog fajla treba ažurirati PVR tabelu dodavanjem događaja novog fajla. Sledi implementacija ovih metoda:

```
@Override
public void eventMediaAdd(PvrEventMediaAdd pvrEventMediaAdd) {
	int mediaIndex = getMediaIndexByTitle(pvrEventMediaAdd.getTitle());
	service.updatePvrData(mediaIndex);
}

@Override
public void eventMediaRemove(PvrEventMediaRemove pvrEventMediaRemove) {
	int mediaIndex = getMediaIndexByTitle(pvrEventMediaRemove.getTitle());
	service.deletePvrData(mediaIndex);
}
```

Kao što je već rečeno, funkcija za brisanje PVR podataka treba da obriše PVR događaje vezane za odgovarajući identifikacioni broj fajla. Sledi funkcija za brisanje PVR podataka:

```
public void deletePvrData(int mediaIndex) {
	getContentResolver().delete(PVR_TABLE_URL,
			PVR_EVENT_FILE_ID + " = " + mediaIndex, null);
}
```

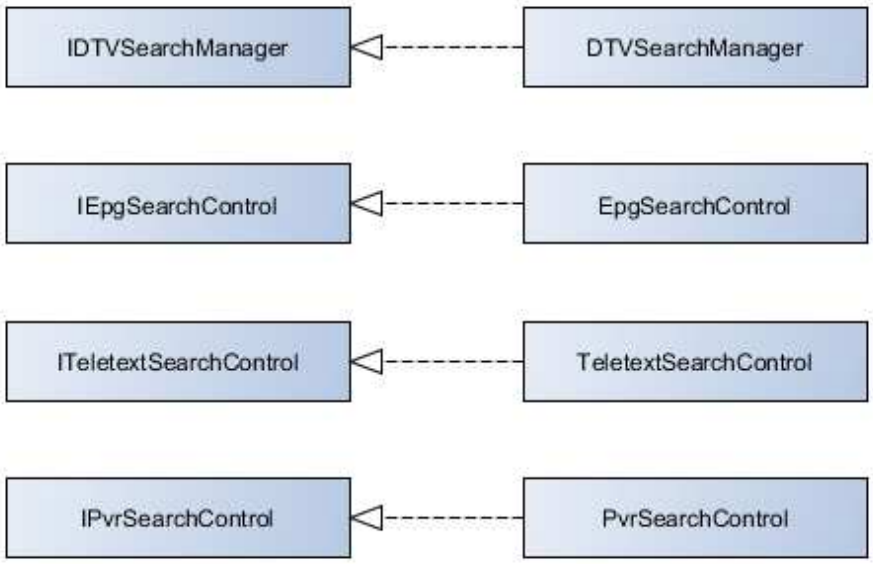
Pored toga što dodaje podatke o događajima vezanim za određeni PVR fajl, funkcija za ažuriranje takođe treba da izračuna „odmaklo“ vreme svakog događaja u PVR fajlu. Ovo vreme se kasnije koristi prilikom reprodukcije ženjenog PVR događaja. Pošto su početna vremena događaja izražena u standardnom UTC vremenu, za vrednost odmaklog vremena za svaki događaj će se koristiti razlika početnog i trenutnog događaja u PVR fajlu. Sledi implementacija funkcije za ažuriranje PVR podataka:

```
public void updatePvrData(int mediaIndex) {
	MetadataInfo metadata = pvrControl.getMetadataInfo(mediaIndex);
	int eventsCount = metadata.getEventsNumber();
	long firstStartTime = UNDEFINED;
	ContentValues values[] = new ContentValues[eventsCount];
	for (int eventIndex = 0; eventIndex < eventsCount; eventIndex++) {
		EpgEvent event = metadata.getEpgEvent(eventIndex);
		long startTime = event.getStartTime().getCalendar()
				.getTimeInMillis();
		if (firstStartTime == UNDEFINED) {
			firstStartTime = startTime;
		}
		ContentValues value = new ContentValues();
		value.put(PVR_EVENT_FILE_ID, mediaIndex);
		value.put(PVR_EVENT_INDEX, eventIndex);
		value.put(PVR_EVENT_DESCRIPTION, event.getDescription());
		value.put(PVR_EVENT_OFFSET_TIME, startTime - firstStartTime);
```

```
            values[eventIndex] = value;
        }
        getContentResolver().bulkInsert(PVR_TABLE_URL, values);
}
```

5.2.2 Realizacija Android4TV API proširenja

Za implementacije definisanog Android4TV API proširenja potrebno je napraviti Java klase koje će realizovati definisane programske sprege. Za četiri prethodno definisane sprežne klase proširenja Android4TV API, realizovane su četiri Java klase. Dijagramom [Slika 5.10] je ilustrovana ova implementacija.



Slika 5.10 Realizacija sprežnih klasa Android4TV API proširenja

Ova implementacija se oslanja na prethodno definisanu bazu podataka, i pretragu inicira formiranjem i upita nad tom formiranom bazom. Od rezultata egzekucije upita se zatim formira lista objekata za odgovarajuću pretragu.

Klasa DTVSearchManager se koristi samo za kontrolu i vođenje evidencije o ostalim klasama proširenog Android4TV API.

Za pretragu EPG podataka, u Java klasi EpgSearchControl realizovana je funkcija getEventsWithText, koja od dobijenih rezultata SQL upita nad EPG tabelom formira listu objekata klase SearchableEpgEvent. Sledi implementacija ove funkcije:

```
@Override
public SearchableEpgEvent[] getEventsWithText(String query) {
        Cursor cursor = context.getContentResolver().query(
                        EPG_TABLE_URL, projection,
                        EVENT_DESCRIPTION + " LIKE '" + query + "%'", null, null);
        SearchableEpgEvent[] events = new SearchableEpgEvent[cursor.getCount()];
        int index = 0;
```

```
	while (cursor.moveToNext()) {
		int serviceIndex = cursor.getInt(service_id);
		int eventIndex = cursor.getInt(event_id);
		EpgEvent event = getRequestedEvent(filterId, serviceIndex, eventIndex);
		events[index++] = new SearchableEpgEvent(event);
	}
	return events;
}
```

Za pretragu teletekst podataka, u Java klasi TeletextSearchControl realizovana je funkcija getPagesWithText, koja od dobijenih rezultata SQL upita nad teletekst tabelom formira listu objekata klase SearchableTeletextPage. Sledi implementacija ove funkcije:

```
@Override
public SearchableTeletextPage[] getPagesWithText(String query) {
	Cursor cursor = context.getContentResolver().query(
			TELETEXT_TABLE_URL, projection,
			PAGE_TEXT + " LIKE '" + query + "%'", null, null);
	SearchableTeletextPage[] events = new
SearchableTeletextPage[cursor.getCount()];
	int index = 0;
	while (cursor.moveToNext()) {
		int trackIndex = cursor.getInt(page_index);
		TeletextTrack track = getTeletextTrack(routeId, trackIndex);
		events[index++] = new SearchableTeletextPage(track);
	}
	return events;
}
```

Za pretragu PVR podataka, u Java klasi PvrSearchControl realizovana je funkcija getEventsWithText, koja od dobijenih rezultata SQL upita nad PVR tabelom formira listu objekata klase SearchablePvrEvent. Sledi implementacija ove funkcije:

```
@Override
public SearchablePvrEvent[] getEventsWithText(String query) {
	Cursor cursor = context.getContentResolver().query(
			PVR_TABLE_URL, projection,
			PVR_EVENT_DESCRIPTION + " LIKE '" + query + "%'", null, null);
	SearchablePvrEvent[] events = new SearchablePvrEvent[cursor.getCount()];
	int index = 0;
	while (cursor.moveToNext()) {
		int mediaIndex = cursor.getInt(file_id);
		int eventIndex = cursor.getInt(event_id);
		MetadataInfo metadata = getMetadataInfo(mediaIndex);
		EpgEvent event = metadata.getEpgEvent(eventIndex);
		events[index++] = new SearchablePvrEvent(event);
	}
```

```
        return events;
}
```

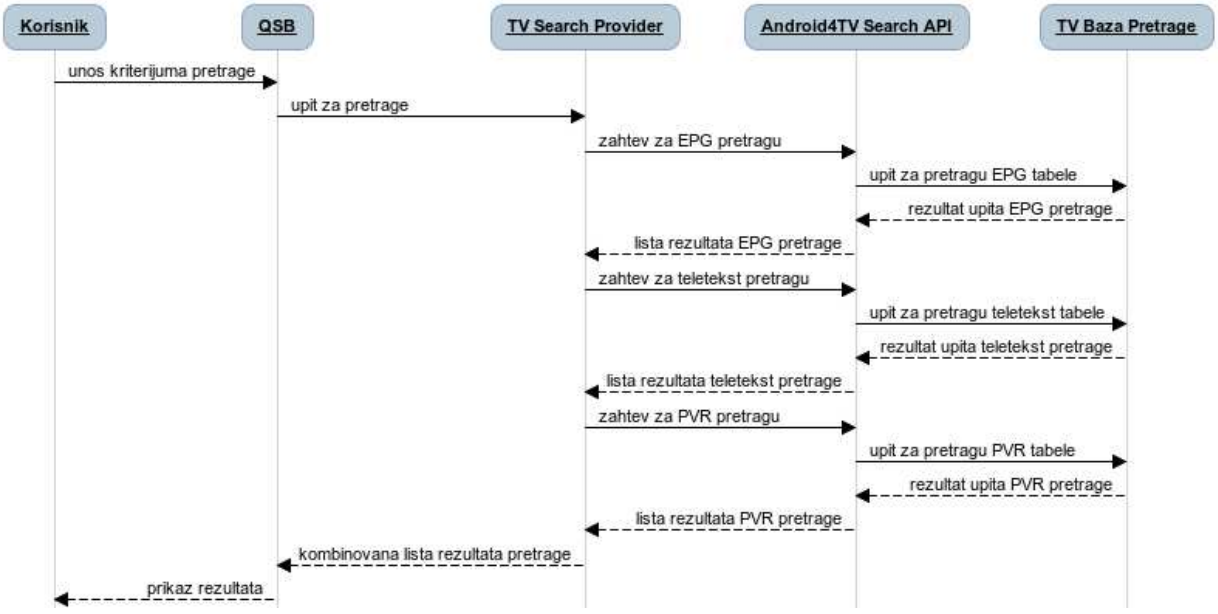
5.3 Implementacija dobavljača podataka pretrage

Android OS obezbeđuje jednostavan mehanizam za proširenje domena pretraživanja QSB aplikacije. Potrebno je implementirati dobavljač podataka (engl. Search Provider) koji se zatim konfiguracionim XML fajlom registruje kao specijalan dobavljač podataka sa pretraživanje. Ovako registrovani dobavljač podataka će za svaki zahtev QSB pretrage biti aktiviran upitom pretrage, a rezultati ovog upita će se naći u listi sugerisanih rezultata QSB pretrage.

Za potrebe implementacije TV pretrage biće napravljen dobavljač podataka pretrage za TV podatke – **TV Search Provider**. Svrha ove aplikacije je da dobavlja podatke TV pretrage i prosleđuje ih QSB aplikaciji. Glavne karakteristike ove aplikacije su:

- Primanje kriterijuma pretrage.
- Preuzimanje rezultata pozivima proširenih Android4TV API funkcija.
- Formiranje rezultata i prosleđivanje istih.

Ilustracijom [Slika 5.11] je prikazan sekvencijalni dijagram korisničkog zahteva za pretragu putem QSB aplikacije, korišćenjem TV Search Provider:



Slika 5.11 Sekvencijalni dijagram obrađivanja zahteva za pretragu

Kada kriterijum pretrage dođe do TV Search Provider dobavljača podataka formira se jedinstvena lista, sastavljena od pronađenih rezultata za sve tri kategorije TV pretrage (EPG, teleteks i PVR). Ova lista se prosleđuje QSB aplikaciji koja će je prikazati krajnjem korisniku.

U narednim poglavljima sledi opis i implementacija pretrage za pomenute kategorije pretrage TV sadržaja.

5.3.1 Pretraga EPG podatak

Nakon proširenja Android4TV API sloja i definisanja svih neophodnih elemenata aplikativne programske podrške za pretragu EPG podataka, implementacija same pretrage u Java aplikativnom domenu se vrlo jednostavno realizuje. Implementacija počinje instanciranjem objekta sprege za kontrolu EPG pretrage, zatim poziva funkcije za inicijaciju pretrage na zadati upit i završava se iteracijom kroz listu rezultata i prosleđivanjem istih do QSB aplikacije. Sledi implementacija funkcije za EPG pretragu u Java programskom domenu:

```
private void getEpgSearchResults(String query, MatrixCursor cursor) {
	IEpgSearchControl epgSearchControl = dtvManager.getEpgSearchControl();

	SearchableEpgEvent events[] = epgSearchControl.getEventsWithText(query);
	for (SearchableEpgEvent event : events) {
		String title = event.getName();
		String suggestion = event.getDescription();
		Uri uri = event.getUri();
		prepareSearchResult(title, suggestion, R.drawable.epg_icon, uri, cursor);
	}
}
```

Pošto se pretraga za EPG sadržajem završi, QSB formira listu rezultata, a svaka stavka u listi je predstavljena sledećim poljima:

- Naslovom rezultata – za EPG rezultat ovo predstavlja naziv emisije.
- Podnaslovom za dodatnu sugestiju rezultata – za EPG rezultat ovo predstavlja kraći opis emisije.
- Sličicom (ikonicom) dobavljača podataka – sličica je uniformna za sve rezultate EPG pretrage.

Takođe, sastavni deo svakog rezultata u QSB listi i URI polje, ali ovo nije vidljivo krajnjem korisniku već se čuva u samoj QSB aplikaciji. Ako korisnik izabere neki od EPG rezultata pretrage, URI polje vezano za taj rezultat poslato je sistemu i odgovarajuća aplikacija treba da odgovori na ovaj poziv, a za URI polje koje generiše EPG pretraga podrazumevana aplikacija je TV aplikacija.

5.3.2 Pretraga teletekst podataka

Implementacija pretrage teletekst sadržaja, korišćenjem navedenih proširenja Android4TV API programske podrške, slična je kao i implementacija EPG pretrage iz prethodnog poglavlja. Prvo treba instancirati objekat sprege za kontrolu teletekst pretrage (ITeletextSearchControl), zatim poziva funkcije za inicijativu pretrage na zadati upit i na kraju iteracijom proći kroz listu

rezultata i proslediti ih do QSB aplikacije. Sledi implementacije funkcije za teletekst pretragu u Java programskom domenu:

```
private void getTeletextSearchResults(String query, MatrixCursor cursor) {
	ITeletextSearchControl ttxSearchControl =
dtvManager.getTeletextSearchControl();

	SearchableTeletextPage pages[] = ttxSearchControl.getPagesWithText(query);
	for (SearchableTeletextPage page : pages) {
		String title = page.getService().getName() + " : " + page.getNumber();
		String suggestion = page.getSearchedTextLine();
		Uri uri = page.getUri();
		prepareSearchResult(title, suggestion, R.drawable.ttx_icon, uri, cursor);
	}
}
```

Pošto se pretraga za teletekst sadržajem završi, QSB formira listu rezultata, a svaka stavka u listi je predstavljena sledećim poljima:

- Naslovom rezultata – za teletekst rezultat ovo predstavlja kombinaciju naziva kanala na kome je teletekst stranica emitovana i broja same stranice.
- Podnaslovom za dodatnu sugestiju rezultata – za teletekst rezultat ovo predstavlja liniju teksta u kojem je traženi upit pronađen.
- Sličicom (ikonicom) dobavljača podataka – sličica je uniformna za sve rezultate teletekst pretrage.

Kao i u slučaju EPG rezultata, sastavni deo svakog teletekst rezultata u QSB listi je i URI polje koje nije vidljivo krajnjem korisniku. I za URI polje koje generiše teletekst pretraga podrazumevana aplikacija je TV aplikacija, tako da kada se URI polje teletekst rezultata pretrage pošalje TV aplikaciji otvara se grafički prikaz teletekst stranice za izabrani rezultat.

5.3.3 Pretraga PVR podataka

Implementacija pretrage PVR sadržaja, korišćenjem navedenih proširenja Android4TV API programske podrške, slična je kao i implementacija EPG i teletekst pretrage iz prethodnih poglavlja. Prvo treba instancirati objekat sprege za kontrolu PVR pretrage (IPvrSearchControl), zatim poziva funkcije za inicijativu pretrage na zadati upit i na kraju iteracijom proći kroz listu rezultata i proslediti ih do QSB aplikacije. Sledi implementacije funkcije za PVR pretragu u Java programskom domenu:

```
private void getPvrSearchResults(String query, MatrixCursor cursor) {
	IPvrSearchControl pvrSearchControl = dtvManager.getPvrSearchControl();

	SearchablePvrEvent files[] = pvrSearchControl.getEventsWithText(query);
	for (SearchablePvrEvent file : files) {
```

```
            String title = file.getInfo().getTitle();
            String suggestion = file.getInfo().getDescription();
            Uri uri = file.getUri();
            prepareSearchResult(title, suggestion, R.drawable.pvr_icon, uri, cursor);
        }
}
```

Pošto se pretraga za PVR sadržajem završi, QSB formira listu rezultata, a svaka stavka u listi je predstavljena sledećim poljima:

- Naslovom rezultata – za PVR rezultat ovo predstavlja naziv emisije unutar PVR snimka.
- Podnaslovom za dodatnu sugestiju rezultata – za PVR rezultat ovo predstavlja naziv same PVR dadoteke, kako i u dialogu za pregled PVR snimaka.
- Sličicom (ikonicom) dobavljača podataka – sličica je uniformna za sve rezultate PVR pretrage.

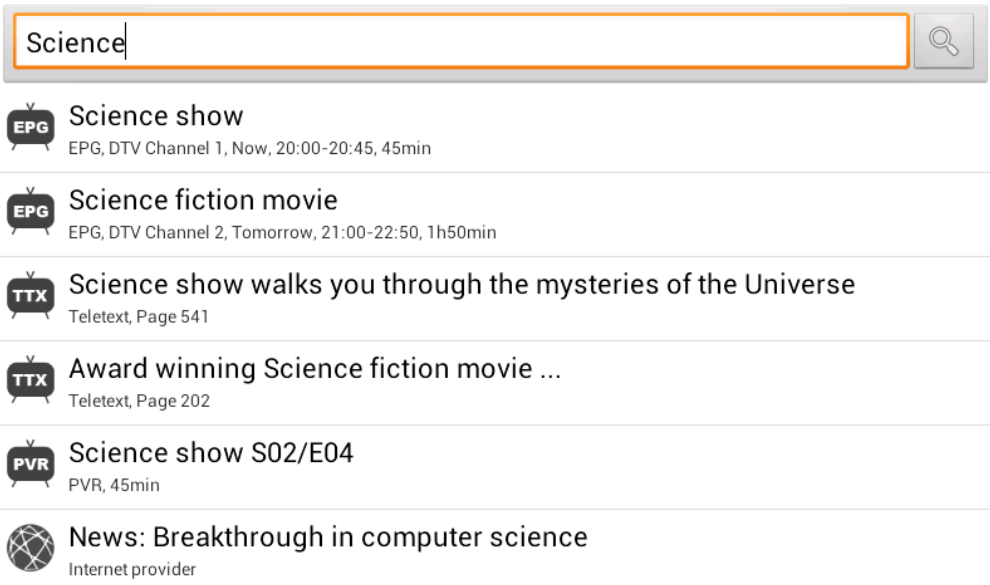
Kao i u slučaju EPG i teletekst rezultata, sastavni deo svakog PVR rezultata u QSB listi je i URI polje koje nije vidljivo krajnjem korisniku. I za URI polje koje generiše PVR pretraga podrazumevana aplikacija je TV aplikacija, tako da kada se URI polje PVR rezultata pretrage pošalje TV aplikaciji, počinje reprodukcija snimka PVR datoteke koja sadrži pronađeni događaj. Takođe vreme početka reprodukcije je početak samog EPG događaja unutar PVR datoteke.

5.3.4 Prikaz i akcije izvršavanje rezultata pretrage

Kao što je već pomenuto, rezultati za sve tri kategorije pretrage formiraju listu rezultata koja se prosleđuje QSB aplikaciji. Kada QSB aplikacija dobije ove rezultate od dobavljača podataka pretrage, oni se sjedinjuju sa rezultatima iz ostalih dobavljača podataka i dobija se jedinstvena lista pretrage za dati kriterijum.

Na ilustraciji [Slika 5.12] prikazan je primer formirane liste rezultata onako kako je grafički prikazuje QSB aplikacija za kriterijum pretrage „Science“.

Svaki od rezultata pretrage definiše jedinstvenu akciju koja će biti izvršena kada korisnik izabere neki od rezultata pretrage. Za rezultate DTV pretrage se, kao što je prethodno opisano, vezuje URI polje i ovo polje se šalje Android sistemu.



Slika 5.12 Grafički prikaz TV rezultata pretrage

5.4 Implementacija aplikacije za odgovor na rezultat pretrage

U cilju eksperimentalne potvrde validnosti rešenja za pretragu TV baziranih podataka, biće implementirana Android aplikacija koja će se pokrenuti kada korisnik izabere rezultat pretrage – **TV Application**. Kao rezultat korisničkog odabira sugerisanog rezultata QSB pretrage, QSB aplikacija će Android sistemu poslati događaj (Intent) sa akcijom za prikaz URI polja dodeljenog tom rezultatu pretrage. Ilustracijom [Slika 5.13] je prikazan sekvencijalni dijagram pokretanja TV Application:

Slika 5.13 Sekvencijalni dijagram startovanja TV aplikacije

Da bi aplikacija reagovala na ovakav događaj, ona mora da se registruje kao podrazumevana aplikacija za prikazivanje URI sheme. Ova registracija se definiše unutar *AndroidManifest.xml* fajla. Sledi izdvojeni deo manifest fajla koje za registraciju na URI shemu:

```
<activity android:name=".TVActivity"
      android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <intent-filter>
```

```
        <action   android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <category android:name="android.intent.category.BROWSABLE" />
        <data android:scheme="tv:// "/>
    </intent-filter>
</activity>
```

Ova aplikacija treba da preduzme odgovarajuće akcije na osnovu tipa izabranog rezultata (EPG, teletekst ili PVR), kao što je to ranije definisano u ovom radu. Aplikacija će se oslanjati na standardne metode Android4TV API za preduzimanje odgovarajuće akcije, što će biti objašnjeno u narednim poglavljima.

5.4.1 Akcija za prikaz EPG rezultata pretrage

Kada se URI podatak EPG rezultata pretrage pošalje TV aplikaciji moguća su dva scenarija reakcija aplikacije:

- Ako je događaj u sadašnjosti (emisija je u toku) aplikacija će prikazati video sadržaj kanala na kome je tražena emisija u toku.
- Ako je događaj u budućnosti (emisija tek treba da bude emitovana) aplikacija će korisniku ponuditi opcije za postavljanje podsetnika kada emisija počne ili automatskog snimanja video sadržaja kada emisija bude emitovana (radi kasnijeg pregleda).

Obradom URI polja dolazi se do informacija o vremenu početka i završetka EPG događaja, kao i o tome koji je identifikacioni broj servisa EPG događaja. Informacija o vremenu događaja je potrebna da bi se odredilo da li je isti u prošlosti ili sadašnjosti, dok se identifikacioni broj servisa koristi za prikazivanje tog servisa ili postavljanje podsetnika za taj servis. Sledi implementacija funkcije za izvršavanje akcije nad pronađenim EPG događajem:

```
public void executeEpgSearchResult(Uri uri) {
       SimpleDateFormat df = new SimpleDateFormat(EPG_DATE_FORMAT);
       Date now = Calendar.getInstance().getTime();

       String startDate = uri.getQueryParameter(EPG_START_DATE);
       String endDate = uri.getQueryParameter(EPG_END_DATE);
       String serviceId = uri.getQueryParameter(EPG_SERVICE_ID);
       if (df.parse(startDate).after(now)) {
              showScheduleDialog(serviceId, startDate, endDate);
       } else {
              dtvManager.getServiceControl().startService(0, 0,
                           Integer.parseInt(serviceId));
       }
```

```
}
```

Treba napomenuti da se EPG događaji iz prošlosti (emisije koje su bile emitovane) ne čuvaju od strane TV programske podrške, te s toga nisu ni pretraživi. Razlog zašto se EPG događaji iz prošlosti ne čuvaju je kako bi se uštedeli memorijski resursi a i skratilo vreme pretrage (sa manje EPG pretraživih podataka kraće je i vreme pretrage). Takođe, tokom izrade ovog rada nije pronađen dovoljno dobar razlog zašto bi korisniku bilo od koristi da pretražuje EPG događaje iz prošlosti, međutim ova mogućnost ostaje otvorena za neka buduća proširenja ove implementacije.

5.4.2 Akcija za prikaz teletekst rezultata pretrage

Za prikaz teletekst rezultata, sve što je potrebno je da se pronađena teletekst stranica prikaže na ekranu, što se postiže pozivom Android4TV API funkcije za ovu akciju. Obradom URI polja pronađenog rezultata dolazi se do broja teletekst stranice koji se kao parametar Android4TV API funkcije koristi za prikaz teletekst stranice. Sledi implementacija funkcije za izvršavanje akcije nad pronađenom teletekst stranicom:

```
public void executeTeletextSearchResult(Uri uri) {
       String track = uri.getQueryParameter(TELETEXT_PAGE_NUMBER);
       dtvManager.getTeletextControl().setCurrentTeletextTrack(0,
                     Integer.parseInt(track));
}
```

5.4.3 Akcija za prikaz PVR rezultata pretrage

Kod prikaza PVR rezultata pretrage, prvo je potrebno startovani pronađeni PVR fajl a zatim premotati video snimak do vremena početka pronađenog EPG događaja unutar PVR fajla. Obradom URI polja dolazi se do identifikacionog broja PVR fajla kao i do odmaklog vremena početka pronađenog događaja. Ova dva podatka su dovoljna za reprodukciju pronađenig PVR događaja.

Sledi implementacija funkcije za izvršavanje akcije nad pronađenim PVR događajem:

```
public void executePvrSearchResult(Uri uri) {
       String fileIndex = uri.getQueryParameter(PVR_FILE_INDEX);
       String timeOffset = uri.getQueryParameter(PVR_TIME_OFFSET);
       dtvManager.getPvrControl().startPlayback(0, Integer.parseInt(fileIndex));
       dtvManager.getPvrControl().jump(0, Integer.parseInt(timeOffset), false);
}
```

5.5 Eksperimentalna potvrda predloženog programskog rešenja

U cilju eksperimentalne potvrde validnosti ponuđenog rešenja, implementirane softverske komponente će biti instalirane na platformi zasnovanoj na Android OS koja podržava Android4TV programsko okruženje. Platforma takođe mora posedovati i TV tjuner kako bi se mogla testirati pretraga podataka koji dolaze putem TV transportnog toka.

U cilju potvrde portabilnosti ponuđenog rešenja, isto je testirano da dve platforme:

- Armada 1500 Pro – proizvod kompanije Marvell
- MPQ8064 – proizvod kompanije Qualcomm.

Karakteristike ovih platformi prikazane su u tabeli [Tabela 5.4].

	Armada 1500 Pro	MPQ8064
SoC	Marvell	Qualcomm
CPU	ARM A9	ARM v7
Takt CPU	1.2 GHz	1.5 GHz
Broj CPU jezgara	4	2
CPU arhitektura	ARM	ARM
RAM	2 GB	2 GB
GPU	Vivante GC4000	Adreno 320
Tjuner	DVB-T	DVB-T/T2/C
Android OS verzija	4.2.2	4.4

Tabela 5.4 Karakteristike testiranih platformi

6. Rezultati merenja

6.1 Rezultati ocene kompletnosti

Za analizu mere kompletnosti rešenja, upoređena su javno dostupna postojećih rešenja sa predloženim rešenjem. Odabir kriterijuma poređenja navedenih rešenja predstavlja neophodan skup funkcionalnosti za realizaciju Android aplikacije za pretraživanje TV baziranog sadržaja. Rešenja koja se porede ovom analizom su:

- Android SDK – zvanično programsko okruženje za izradu Android aplikacija, vlasništvo kompanije Google. Detalji ovog rešenja mogu se videti u [Android SDK].
- Java TV – proširenje standardne Java podrške za izradu TV aplikativnog softvera, izdato od kompanije Oracle. Detalji ovog rešenja mogu se videti u [Java TV].
- Android4TV API – aplikativna programska sprega za izradu TV aplikacija u Android OS, izdato od kompanije iWedia. Detalji ovog rešenja mogu se videti u [Android4TV].
- Android4TV Search API – proširenje Android4TV API predstavljeno u ovom radu.

U tabeli [Tabela 6.1] prikazani su rezultati analize poređenih rešenja radi utvrđivanja mere kompletnosti datog rešenja.

Funkcionalnost	Android SDK	Java TV	Android4TV API	Android4TV Search API
Reprodukcija MPEG video sadržaja	✓	✗	✓	✓ 1
Reprodukcija DTV sadržaja	✗	✗	✓	✓ 1
Pristup listi kanala	✗	✓	✓	✓ 1
Pristup EPG sadržaju	✗	✓	✓	✓ 1

Pretraga EPG sadržaja	✗	✗	✗	✓
Postavljanje EPG alarma	✗	✓	✓	✓ [1]
Prikazivanje teletekst sadržaja	✗	✗	✓	✓ [1]
Pristup teletekst podacima	✗	✗	✓	✓ [1]
Pretraga teletekst podataka	✗	✗	✗	✓
Snimanje DTV sadržaja (PVR)	✗	✗	✓	✓ [1]
Reprodukcija PVR sadržaja	✗	✗	✓	✓ [1]
Pristup PVR lisi snimaka	✗	✗	✓	✓ [1]
Pretraga PVR sadržaja	✗	✗	✗	✓
	[1] Android4TV Search API nasleđuje ove funkcije od Android4TV API			

Tabela 6.1 Rezultati analize za meru kompletnosti

Analizom rezultata dobijenih poređenjem predloženog rešenja programske sprege za pretragu TV baziranih servisa sa navedenim postojećim rešenjima za Android OS može se zaključiti da je predloženo rešenje najkompletnije.

6.2 Rezultati ocene ažurnosti

Za analizu ažurnosti rešenja pretrage, ponuđeno rešenje je upoređeno sa Google TV komercijalnim rešenjem. Ključna razlika ovih rešenja je što Google TV podatke dobavlja preko interneta dok implementirano rešenje u ovog rada podatke dobavlja iz digitalnog TV transportnog toka podataka.

U tabeli [Tabela 6.2] prikazana je analiza mere ažurnosti za ova dva rešenja.

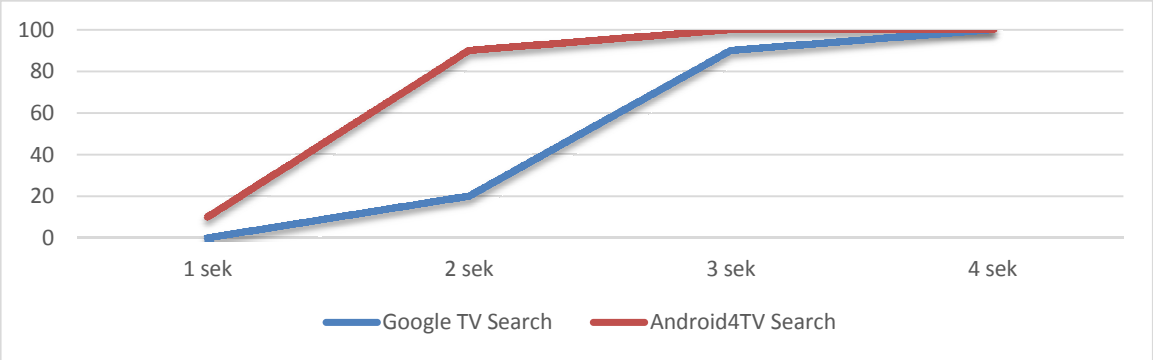
Proteklo vreme nakon promene EPG događaja	Google TV	Android4TV
1	0/10	1/10
2	2/10	9/10
3	9/10	10/10
4	10/10	10/10
Vrednosti su izražene u minutima	Vrednosti su izražene u broju pronađenih rezultata u odnosu na broj iniciranih pretraga	

Tabela 6.2 Rezultati analize za meru ažurnosti

Tok ove analize podrazumeva se da kada dođe do promene u tabeli emitovanih programa od strane emitera izvrši pretraga sa određenim zakašnjenjem i vidi da li je promenjeni događaj

deo rezultata pretrage. Naravno kriterijum pretrage mora biti takav da će se promenjeni događaj sigurno naći u listi rezultata. Rezultati su se merili za pretrage sa analizom za jednu, dve, tri i četiri sekunde a svaki put je pretraga izvršena po 10 puta.

Na grafiku [Slika 6.1] su prikazane procentualne vrednosti ove analize; na apscisi je prikazano proteklo vreme nakon promene EPG događaja (u sekundama), a na ordinati je prikazan odnos pronađenih događaja od broja pokušaja (u procentima).



Slika 6.1 Grafik analize za meru ažurnosti

Analizom dobijenih rezultata može se zaključiti da su podaci pretrage informacija iz digitalnog TV transportnog toka ažurniji neposredno nakon promene podataka, međutim vremenom se ažurnost izjednačuje.

6.3 Rezultati ocene performansi

Mera performansi treba da pokaže kakvo će biti korisničko iskustvo prilikom korišćenja pretrage u smislu brzine prikazivanja rezultata. Implementirano rešenje je i ovde poređeno sa Google TV čija se pretraga zasniva na internet servisima. U cilju dobijanja realnije slike rezultata, poređenje je izvršeno nad Google TV sistemom za različite vrednosti internet protoka (10 Mbit/s, 1 Mbit/s i 0.1 Mbit/s).

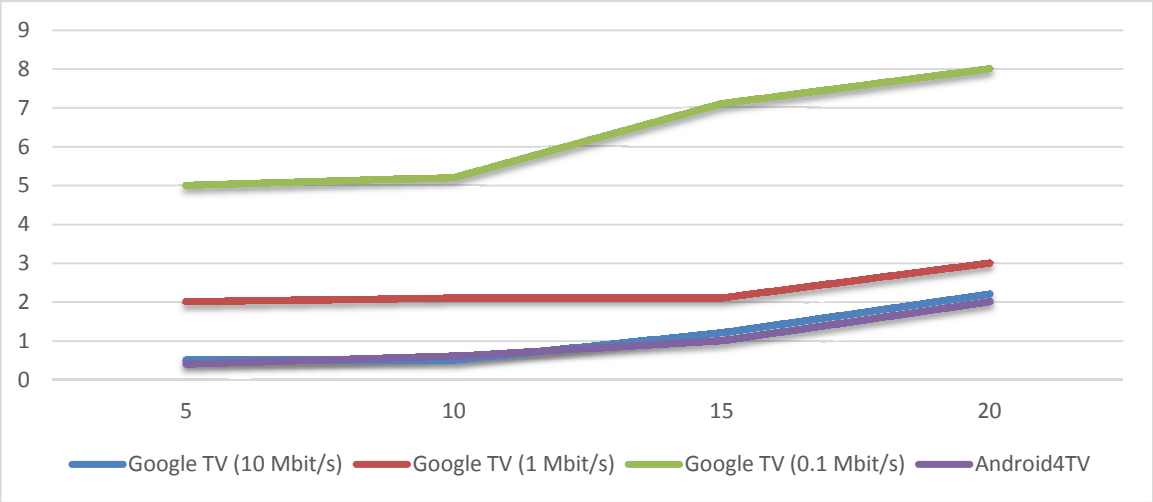
Rezultati ove analize se nalaze u tabeli [Tabela 6.3].

Broj pronađenih rezultata	Google TV (10 Mbit/s)	Google TV (1 Mbit/s)	Google TV (0.1 Mbit/s)	Android4TV
5	0,5	2,0	5,0	0,4
10	0,5	2,1	5,2	0,6
15	1,2	2,1	7,1	1,0
20	2,2	3,0	8,0	2,0

Tabela 6.3 Rezultati analize za meru performansi

Prilikom ove analize merilo se vreme od trenutka kada korisnik inicira pretragu za traženu ključnu reč do trenutka kada se svi rezultati prikazu na ekranu. Pretraga za istu ključnu reč (sa istim brojem pronađenih rezultata) se testirala za svako rešenje.

Ilustracija [Slika 6.2] grafički prikazuje vrednosti ove analize; na apscisi je prikazan broj pronađenih rezultata inicirane pretrage, a na ordinati je predstavljeno proteklo vreme od iniciranja pretrage do dobijanja liste rezultata.



Slika 6.2 Grafik analize za meru performansi

Kao što se može videti iz dobijenih rezultata analize, oba rešenja pokazuju veoma slične performanse kada je internet konekcija kvalitetna i stabilna. Međutim, smanjenjem protoka internet podataka, Google TV rešenje znatno gubi na performansama i u takvim uslovima rešenje predstavljeno u ovom radu znatno brže prikazuje rezultate pretrage.

7. Zaključak

Problematika kojom se ovaj rad bavi je kako integrisati pretragu TV baziranih podataka u uređaj zasnovan na Android OS. Za pokretanje pretrage od strane korisnika iskorišćena je QSB standardna Android aplikacija i rezultati pretrage ove aplikacije su prošireni rezultatima TV pretrage za EPG, teletekst i PVR. U cilju rešavanja problema implementacije napravljeno je proširenje za Android4TV razvojno okruženje, tako da je sada moguće i pretraživati TV bazirane podatke. Na ovaj način je pretraga TV sadržaja lako izvodiva iz Android aplikativnog domena. Ovoj mehanizma pretraživanja upotrebljen je za dobavljanje TV podataka prilikom QSB pretrage. Integracijom pretrage u QSB aplikaciju, TV uređaj ravnopravno tretira pretragu internet i TV podataka, što krajnjem korisniku odaje utisak kompaktnosti rešenja.

Kompletno programsko rešenje, opisano u ovom radu, za integraciju TV bazirane pretrage sastoji se od četiri programska modula:

- Android4TV Search API – Java aplikativna programska sprega koja proširuje mogućnosti javne Android4TV API programske sprege, dodajući funkcionalnost pretraživanja EPG, teletekst i PVR sadržaja.
- TV Search Service – Android aplikativni servis koji koristeći se Android4TV API formira i održava bazu podataka nad kojom se vrši pretraga TV baziranih podataka. Implementacija Android4TV Search API neposredno koristi ovu bazu podataka kao izvor informacija pretraživih podataka.
- TV Search Provider – Android dobavljač podataka koji se neposredno koristi od strane QSB aplikacije kao izvor informacija. Ovaj dobavljač podataka se oslanja na Android4TV Search API za iniciranje funkcija za pretraživanje i njihove rezultate prosleđuje QSB aplikaciji..
- TV Application – Android aplikacija koja se odaziva na odabrane TV rezultate QSB pretrage i preduzima odgovarajuće akcije za svaku kategoriju rezultata (EPG,

teletekst ili PVR). Ova aplikacija se oslanja na Android4TV API za izvršavanje odgovarajućih TV relevantnih akcija.

Ideja za uvođenjem pretrage TV sadržaja nije nova, i postoje mnoga komercijalna rešenja kao i radovi na ovu temu. U ovom radu je opisano rešenje implementirano koristeći samo Java programski jezik i javne besplatno dostupne biblioteke za programsku podršku. Na ovaj način je dobijeno rešenje portabilno i nezavisno od sistemskih proširenja. Portabilnost ponuđenog rešenja je i eksperimentalno potvrđena testiranjem na dve platforme različitih proizvođača i karakteristika:

- Armada 1500 Pro – kompanije Marvell
- MPQ8064 – kompanije Qualcomm.

Dalji razvoj može obuhvatiti proširenje domena pretrage sa EPG, teletekst i PVR na ostale DTV sadržaje kao što su: MHEG-5, HbbTV ili DTV Subtitles. Međutim, kako Android4TV API trenutno ne podržava direktan pristup ovim informacijama, za implementaciju pretrage ovih podataka bilo bi neophodno proširiti i samo Android4TV programsko rešenje.

Još jedna mogućnost daljeg razvoja ovog rešenja moglo bi se odnositi na praćenje korisničkih navika kroz analizu iniciranih pretraga na TV uređaju. Na ovaj način se može poboljšati kvalitet servisnih usluga TV uređaja tako što će se korisniku preporučiti sadržaj koji je na osnovu analize čestih pretraga najbolje ocenjen, i samim tim najverovatnije omiljen kod korisnika TV uređaja.

8. Literatura

[A4TV App] Android4TV Aplikacija, https://github.com/iWedia/android4tv-app-v2

[Abe] K. Abe and N. Sugita, "*Distances between strings of symbols-Review and remarks*", Proc. ICPR, pp. 172-174, 1982.

[Ableson] F. Ableson, C. Collins, R. Sen, "*Unlocking Android*," Manning Publications Co., Greenwich, CT, USA, 2009.

[Android SDK] Android SDK, http://developer.android.com/sdk/index.html

[Android4TV] Android4TV SDK, http://android4tv.iwedia.com/

[AndroidSDK] Android SDK, http://developer.android.com/sdk/index.html

[Bjelic] V. Bjelić, „*Unapređenje teletekst podrške na prijemniku digitalnog televizijskog signala zasnovanog na Android operatisnom sistemu*", bečler rad, Novi Sad 2013

[Brandao] R.R. de Mello Brandao, G.L. de Souza Filho, C.E.C.F. Batista, and L.F. Gomes Soares, "Extended Features for the Ginga- NCL Environment: Introducing the LuaTV API," 19th International Conference on Computer Communications and Networks (ICCCN), Zurich, August 2010, pp. 1-6.

[Djukic] I. Djukic, N. Lukic, R. Dzakula, D. Srdjan, „*A Java API interface for the search of the EPG data in Android OS based devices*"

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

[GoogleTV] GoogleTV, http://www.google.com/tv/

[Java TV] Java TV, http://www.oracle.com/technetwork/java/javame/

javatv/overview/getstarted/index.html

[Kovacevic] B. Kovačević, „*Једно решење реализације и приказа функционалности дигиталног снимача на пријемнику дигиталног телевизијског сигнала заснованом на Андроид платформи*", Master Rad, Fakultet Tehničkih Nauka Novi Sad 2014

[Kuzmanovic] N. Kuzmanovic, T. Maruna, M. Savic, G. Miljkovic, D. Isailovic, "*Google's Android as an application environment for DTV decoder system*," 14th IEEE International Symposium on Consumer Electronics (ISCE), June 2010, pp. 1-5.

[Lu] Y. S. Lu, C. H. Lee, H. Y. Weng, Y. M. Huang, "*Design and implementation of digital TV widget for Android on multi-core platform*", International Computer Symposium (ICS), December 2010, pp. 576-580.

[Lucas] A. dos Santos Lucas, S.D. Zorzo, "*Personalization for Digital Television Using Recommendation System strategy*", 16th International Conference on Systems, Signals and Image Processing (IWSSIP), Chalkida, June 2009, pp. 1-4.

[Lukic] N. Lukić, *"Predlog proširenja Android operativnog sistema servisima digitalne televizije"*, doktorska disertacija, Novi Sad 2014

[Lukic2] N. Lukic, N. Teslic, T. Maruna, V. Mihic, *"A java API interface for the search of DTV services in embedded multimedia devices,"* Consumer Electronics, IEEE Transactions on, vol.59, no.4, pp.875,882, November 2013

[OpenGL|ES] OpenGL ES standard, https://www.khronos.org/opengles/

[Vidakovic] 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.

[Vidakovic2] M. Vidakovic, T. Maruna, N. Teslic, V. Mihic, "*A java API interface for the integration of DTV services in embedded multimedia devices*," Consumer Electronics, IEEE Transactions on, vol.58, no.3, pp.1063,1069, August 2012.

[WebKit] The WebKit Open Source Project, www.webkit.org/