



УНИВЕРЗИТЕТ У НОВОМ САДУ ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



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

ЗАВРШНИ (BACHELOR) РАД

Кандидат: Бранимир Ковачевић
Број индекса: Е12686

Тема рада: ЈЕДНА РЕАЛИЗАЦИЈА ЈАВА СЕРВИСА ЗА КОМУНИКАЦИЈУ
ИЗМЕЂУ ПРОГРАМСКЕ ПОДРШКЕ ДИГИТАЛНОГ
ТЕЛЕВИЗИЈСКОГ ПРИЈЕМНИКА И ЈАВА АПЛИКАЦИЈЕ У
АНДРОИД ОКРУЖЕЊУ

Ментор рада: Проф. др Никола Теслић

Нови Сад, јул, 2012.



КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:	
Идентификациони број, ИБР:	
Тип документације, ТД:	Монографска документација
Тип записа, ТЗ:	Текстуални штампани материјал
Врста рада, ВР:	Завршни (Bachelor) рад
Аутор, АУ:	Бранимир Ковачевић
Ментор, МН:	Проф. др Никола Теслић
Наслов рада, НР:	ЈЕДНА РЕАЛИЗАЦИЈА ЈАВА СЕРВИСА ЗА КОМУНИКАЦИЈУ ИЗМЕЂУ ПРОГРАМСКЕ ПОДРШКЕ ДИГИТАЛНОГ ТЕЛЕВИЗИЈСКОГ ПРИЈЕМНИКА И ЈАВА АПЛИКАЦИЈЕ У АНДРОИД ОКРУЖЕЊУ
Језик публикације, ЈП:	Српски / латиница
Језик извода, ЈИ:	Српски
Земља публиковања, ЗП:	Република Србија
Уже географско подручје, УГП:	Војводина
Година, ГО:	2012
Издавач, ИЗ:	Ауторски репринт
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, ФО: (поглавља/страна/ цитата/табела/слика/графика/прилога)	7/35/0/0/11/0/0
Научна област, НО:	Електротехника и рачунарство
Научна дисциплина, НД:	Рачунарска техника
Предметна одредница/Кључне речи, ПО:	Телевизија, програмска подршка телевизијског пријемника, Јава сервис, Андроид
УДК	
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, ВН:	
Извод, ИЗ:	У овом раду је приказано једно решење Јава сервиса који користи Comedia програмску подршку за дигиталну телевизију. Решење је реализовано на Marvell BG2 SOC платформи са сателитским бирачем канала. Такође је приказан пример апликације која користи Јава сервис као комуникацију са Comedia програмском подршком.
Датум прихватања теме, ДП:	
Датум одбране, ДО:	
Чланови комисије, КО:	Председник: Проф. др Јелена Ковачевић
Члан:	
Члан, ментор:	Проф. др Никола Теслић
	Потпис ментора



KEY WORDS DOCUMENTATION

Accession number, ANO:			
Identification number, INO:			
Document type, DT:	Monographic publication		
Type of record, TR:	Textual printed material		
Contents code, CC:	Bachelor Thesis		
Author, AU:	Branimir Kovačević		
Mentor, MN:	PhD Nikola Teslić		
Title, TI:	One soution of Java service used for communication between digital television middleware layer and Java application developed in Android		
Language of text, LT:	Serbian		
Language of abstract, LA:	Serbian		
Country of publication, CP:	Republic of Serbia		
Locality of publication, LP:	Vojvodina		
Publication year, PY:	2012		
Publisher, PB:	Author's reprint		
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6		
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/applications)	7/35/0/0/11/0/0		
Scientific field, SF:	Electrical Engineering		
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems		
Subject/Key words, S/KW:	Television, Middleware, Android, Java service		
UC			
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia		
Note, N:			
Abstract, AB:	This paper presents a solution of Java service that uses Comedia middleware for digital television. Solution is implemented on Marvell BG2 SOC platform with satellite tuner. Solution also shows an example application that uses Java service for communicating with Comedia digital television middleware.		
Accepted by the Scientific Board on, ASB:			
Defended on, DE:			
Defended Board, DB:	President:	Prof. dr Jelena Kovačević	
	Member:		Menthor's sign
	Member, Mentor:	Prof. dr Nikola Teslić	

Zahvalnost

Za nesebičnu posvećenost i strpljenje prilikom izrade završnog (Bachelor) rada zahvalnost dugujem mentoru, prof. dr Nikoli Tesliću, redovnom profesoru Fakulteta tehničkih nauka Univerziteta u Novom Sadu.

Podjednaku zahvalnost na korisnim informacijama, sugestijama i savetima dugujem Tomislavu Maruni.

Posebnu zahvalnost dugujem porodici na podršci i devojci Miljani na višednevnom lektorisanju rada.

U Novom Sadu, jula 2012. godine

Branimir Kovačević

SADRŽAJ

1.	Uvod.....	1
2.	Teorijske osnove.....	3
3.	Koncept rešenja	6
3.1	Ključne komponente sistema	7
4.	Programsko rešenje.....	10
4.1	Tipovi Android servisa i njihove specifičnosti	10
4.2	Povezivanje i raskidanje veze sa Android servisom	11
4.3	Obaveštenja i povratne metode iz Android servisa.....	11
4.4	Prenos složenih klasa između različitih procesa	13
4.5	Moduli i metode Android servisa.....	13
4.5.1	Paket android.os.....	14
4.5.2	Paket com.android.server	14
4.5.3	Paket android.dtv.pvr	15
4.5.4	Paket android.dtv.reminder.....	15
4.5.5	Paket android.dtv.service.....	15
4.5.6	Paket android.dtv.setup	16
4.5.7	Paket android.dtv.audio.....	16
4.5.8	Paket android.dtv.epg	16
4.5.9	Paketi android.dtv.mheg, android.dtv.subtitle, android.dtv.teletext	17
4.5.10	Paket android.dtv.video	17
4.5.11	Paket android.dtv.ca	17
5.	Ispitivanje i verifikacija	19
5.1	Ručno pretraživanje kanala.....	20
5.2	Prebacivanje kanala	21

5.3	Prikaz informacija o kanalu pozivom povratne metode	22
5.4	Promena više zvučnih tokova na jednom kanalu	23
5.5	Elektronski programski vodič	24
6.	Zaključak	25
7.	Literatura	27

SPISAK SLIKA

Slika 2.1 Android arhitektura sa DTV dodatkom.....	3
Slika 2.2 Komunikacija Java aplikacije sa programskom podrškom televizijskog prijemnika.....	5
Slika 3.1 Slojevi programske podrške zasnovane na Android servisu	7
Slika 3.2 Komponente Android DTV servisa	8
Slika 3.3 Struktura komponente IServiceListControl.....	9
Slika 4.1 Implementacija sistema obaveštenja.....	12
Slika 5.1 Ručno pretraživanje kanala	20
Slika 5.2 Prebacivanje kanala.....	21
Slika 5.3 Prikaz informacija o trenutnom kanalu	22
Slika 5.4 Prikaz liste dostupnih zvučnih tokova na trenutnom kanalu	23
Slika 5.5 Prikaz elektronskog programskog vodiča	24

SKRAĆENICE

- AIDL** - *Android Interface Definition Language*, Android jezik za definisanje sprege
- CAM** - *Conditional – Access module*, Modul za uslovni pristup
- DTV** - *Digital Television*, Digitalna televizija
- IPC** - *Inter-process communication*, međuprocesna komunikacija
- JNI** - *Java native interface*, sprega Java programskog jezika i C koda
- OS** - *Operating system*, Operativni sistem

1. Uvod

U ovom radu je prikazano jedno rešenje Java servisa koji komunicira sa Comedia sprežnim slojem programske podrške (eng. Middleware) proizvođača iWedia [1]. Rešenje je realizovano na Marvell BG2 SOC platformi sa satelitskim biračem kanala (eng. *Tuner*).

Kao ključno pitanje prilikom razvoja moderne programske podrške za digitalnu televiziju nameće se pitanje koji programski jezik koristiti. Prilikom razvoja programske podrške za Android operativni sistem, prirodno okruženje za razvoj programeru predstavlja Java programski jezik. Ovaj jezik obično zahteva fizičke arhitekture sa više resursa, što često nije slučaj sa sistemima u realnom vremenu. Da bi iskoristili sve dostupne resurse, programeri koriste programske jezike poput C/C++, ali ovim rešenjima nedostaje olakšana prenosivost koda i pojednostavljenje otklanjanje grešaka [2]. U poslednje vreme, platforme za rad u realnom vremenu dobijaju sve više procesorske snage sa više procesorskih jezgara i većim radnim taktom. Iz tog razloga moguće je koristiti programske jezike poput Java radi zadovoljenja većine potreba. Pojedini poslovi se i dalje moraju obavljati u programskom okruženju C/C++, a sve u cilju približavanja granici mogućnosti same fizičke arhitekture.

U okviru ovoga rada dat je pregled načina prenosa podataka između procesa, kao i strukture podataka koje se prenose između procesa. Takođe je prikazan način komunikacije Java servisa sa Comedia programskom podrškom korišćenjem sprege Java programskog jezika i C koda (eng. *JNI - Java native interface*).

Rezultati su upoređivani korišćenjem prilagođene Java aplikacije koja koristi Java servis za dobavljanja podataka od Comedia programske podrške.

Ovaj rad je sačinjen od sedam poglavlja.

Drugo poglavlje opisuje prednosti i razloge korišćenja Java servisa u odnosu na trenutno rešenje programske podrške televizijskog prijemnika.

U trećem poglavlju dat je pregled slojeva sistema koji u sebe uključuju Android servis kao i opis ključnih komponenti sistema.

Četvrto poglavlje sadrži detaljan opis uvedenog Android servisa, prikaz sprege Java i C koda, kao i način korišćenja servisa u Android aplikaciji.

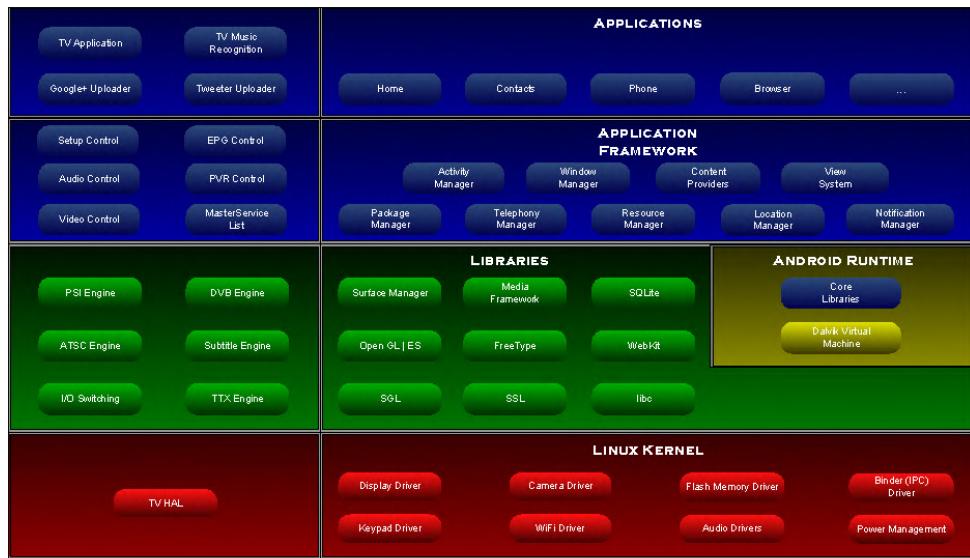
U petom poglavlju su opisani načini ispitivanja i verifikovanja svih modula u Java servisu kao i sprege Java programskog jezika i C koda.

Šesto poglavlje sadrži kratak pregled onoga što je urađeno u ovom radu i kakvi su dalji pravci razvoja.

U sedmom poglavlju dat je spisak korišćene literature za izradu rada.

2. Teorijske osnove

Android predstavlja programsku podršku prvenstveno za mobilne uređaje koji u sebe uključuje operativni sistem, sprežni sloj za povezivanje Java i C programskog jezika i pojedine aplikacije koje su dostupne korisnicima. Sistem je zasnovan na Linux jezgru. Android predstavlja trenutno najrasprostanjeniji sistem za mobilne uređaje, ali su sve češće dostupna određena prilagođenja Android operativnog sistema za druge uređaje koji rade u realnom vremenu poput: tablet računari, televizijski prijemnici i drugi. Na slici 2.1 je dat prikaz većih komponenti Android operativnog sistema.



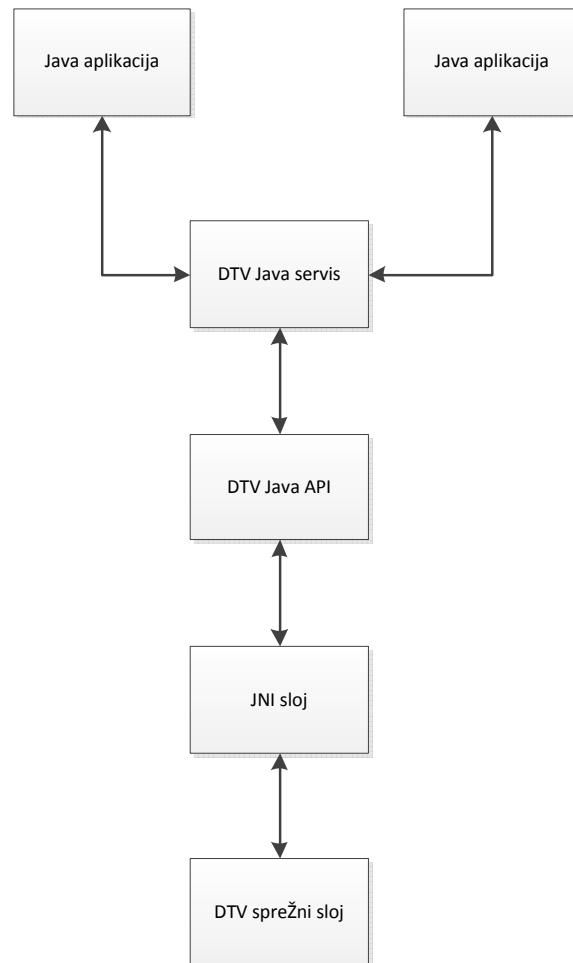
Slika 2.1 Android arhitektura sa DTV dodatkom

Na postojeće slojeve Android operativnog sistema dodat je deo programske podrške za digitalne televizijske prijemnike da bi se postojeća rešenja programske podrške televizijskog prijemnika mogla koristiti u Android OS. Da bi druge Android aplikacije(osim osnovne televizijske aplikacije) mogle koristiti informacije o trenutnom kanalu, elektronskom

programskom vodiču i slično, potrebno je uvesti pojam Java servisa koji komunicira kao posrednik između programske podrške televizijskog prijemnika i svake druge Android aplikacije koja se sa servisom povezuje. Bez uvođenja servisa, u jednom trenutku, samo jedna aplikacija može da koristi i komunicira sa programskom podrškom televizijskog prijemnika.

Prednost Android servisa je u tome što u Android okruženju servis može da obavlja posao u pozadini, čak i dok je korisnik u drugoj aplikaciji. Servis dozvoljava drugim aplikacijama da se povežu sa njim, da razmenjuju podatke i obavljaju međuprocesnu komunikaciju. Još jedna prednost je u tome što više korisničkih aplikacija može istovremeno biti povezano na servis, slati mu zahteve i prihvataći rezultate obrade.

Da bi se povezale korisničke aplikacije sa servisom potrebno je definisati spregu kojom se odvija komunikacija između klijenta i servisa. Definisanje sprege se obavlja uz pomoć odgovarajućeg jezika stvaranog za tu namenu (eng. AIDL - *Android Interface Definition Language*) [3]. Na Androidu jedan proces ne može jednostavno da pristupi memoriskom prostoru drugog procesa. AIDL obavlja posao rastavljanja složenih objekata na primitivne tipove podataka, koje operativni sistem razume i koje može da šalje između različitih procesa. Svaki složeni objekat koji se prenosi mora da podržava protokol rastavljanja složenih tipova na proste i sastavljanja prostih tipova u složen (eng. *Parcelable protocol*, u daljem tekstu korišćen je izraz parselabilan protokol), kako bi objekat mogao biti rastavljen na primitivne tipove i prenet između procesa. Nakon obrade složenog objekta, ili primanja zahteva od klijenta, isti je potrebno proslediti programskoj podršci televizijskog prijemnika. Programska jezik Java prevodi se u bajt-kod. Ovaj kod se izvršava u Java virtuelnoj mašini. Za razliku od Java, drugi programski jezici se prevode u mašinski kod koji se direktno izvršava u procesoru. Neophodno je prilagoditi deo koda, napisanog u drugom programskom jeziku, tako da se može izvršiti iz Java programa. Programska podrška televizijskog prijemnika je napisana u C kodu, a servis u Java kodu, pa se koristi JNI sprega [4] da bi se obezbedilo pozivanje C koda i prosleđivanje podataka C kodu iz Java programa. Celokupna komunikacija je ilustrovana na slici 2.2.



Slika 2.2 Komunikacija Java aplikacije sa programskom podrškom televizijskog prijemnika

Postojeće rešenje programske podrške proizvođača iWedia, na kome se zasniva i ovaj rad, predlaže korišćenje televizijske aplikacije kao osnovne aplikacije (kao što je mobilna aplikacija za razgovore na mobilnom telefonu) sa kojom započinje i završava se korišćenje televizora. Ne samo osnovna televizijska aplikacija, nego i sve aplikacije koje budu razvijane u budućnosti, moći će da koriste standardizovanu programsku spregu (eng. *API – Application programming interface*) izloženu kroz Android servis bez potrebe za dodatnim bibliotekama i pisanja C koda.

3. Koncept rešenja

Integracija digitalne televizije u Android operativni sistem zahteva postojanje više slojeva programske podrške. Svaki od slojeva predstavlja prilagođen servis za druge slojeve.

Na najvišem nivou nalaze se Android aplikacije koje obavljaju grafičko prikazivanje videa, prikazivanje elektronskog programskog vodiča, liste dostupnih kanala i drugih informacija značajnih za televizijski servis. Različite televizijske aplikacije se od sada mogu pisati na identičan način kao i ostale Android aplikacije - korišćenjem standardnog Android okruženja.

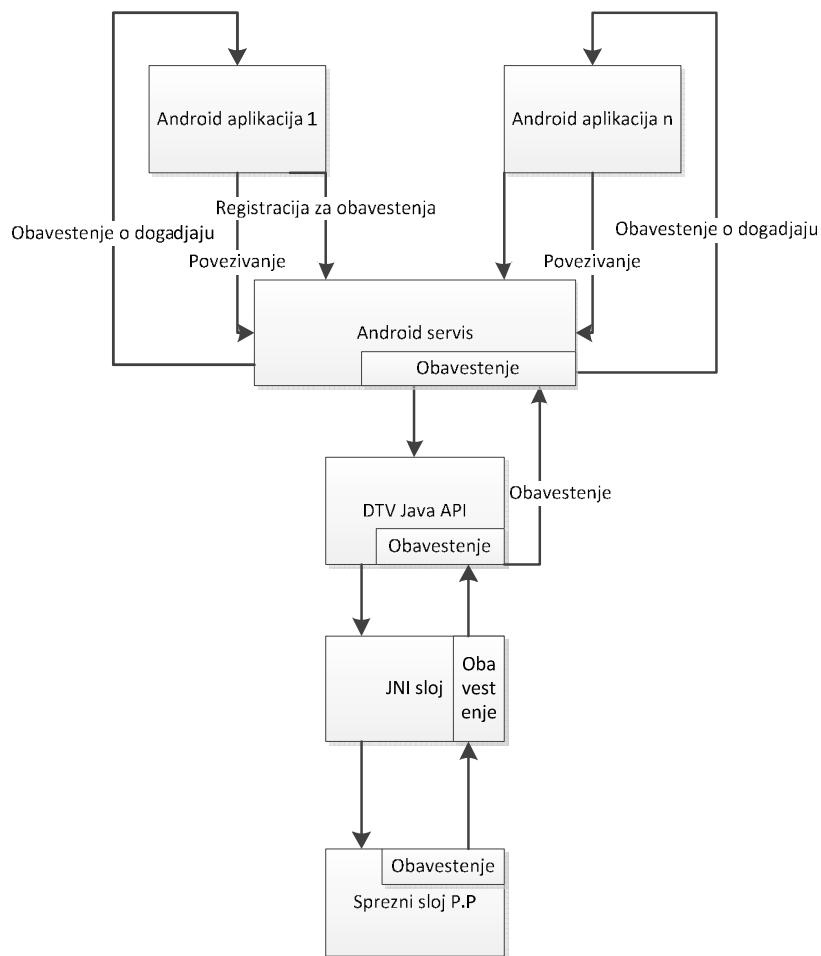
Sledeći sloj čini Android servis koji omogućava razmenu podataka između Android aplikacija i programske podrške televizijskog prijemnika. Postojanjem servisa programer više ne mora da bude upoznat sa realizacijom programske podrške televizijskog prijemnika.

Android servis naleže na skup funkcija koje iz Java korisničkog prostora komuniciraju sa programskom podrškom televizijskog prijemnika. Spregu između ova dva sloja predstavlja JNI sloj.

Ispod toga sloja nalazi se sloj programske podrške televizijskog prijemnika, specifičan za svakog proizvođača, koji oni mogu da prilagode tako da najviše odgovara njihovim potrebama.

Ukoliko se desi neki asinhron događaj karakterističan za televizijski prijemnik, a za čije promene se korisnik prijavio, programska podrška obaveštava JNI sloj koji poziva određenu metodu Android servisa. Ta metoda, po potrebi, javlja prijavljenim aplikacijama na servis da se promena određenog tipa upravo desila.

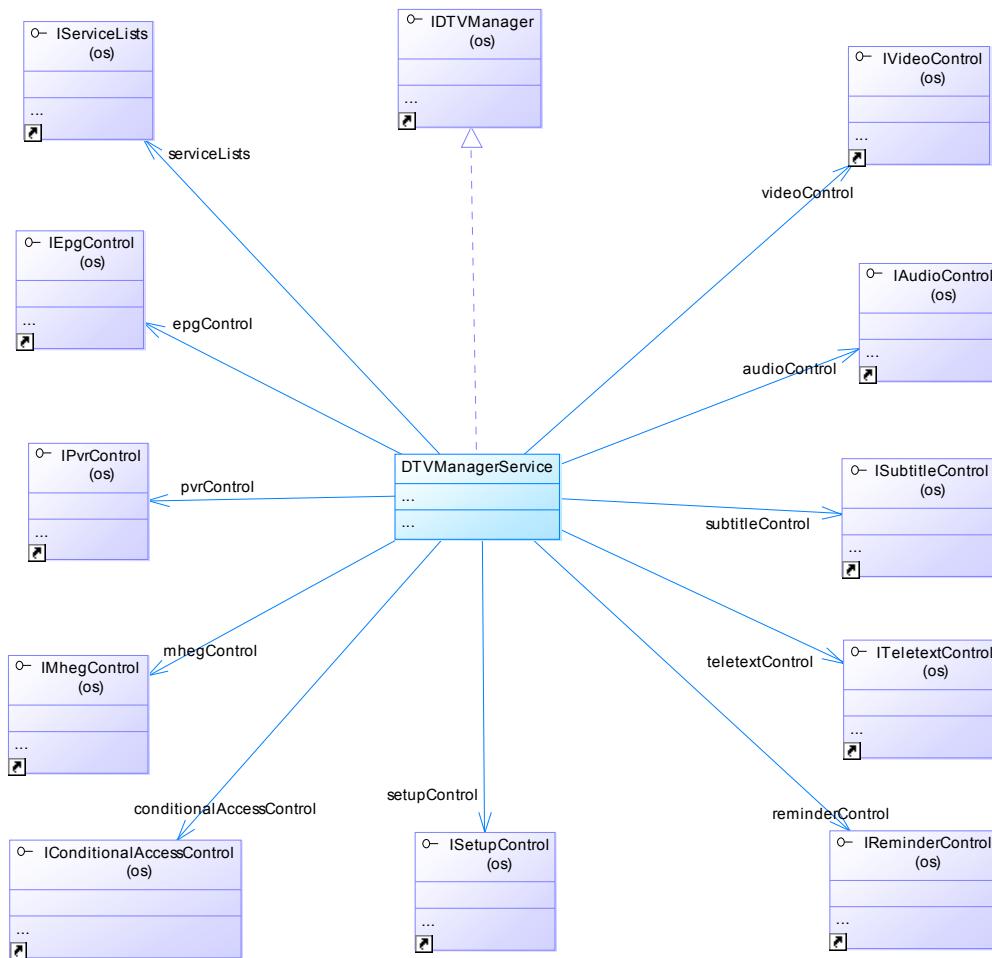
Detaljnije predstavljanje slojeva je dato na slici 3.1.



Slika 3.1 Slojevi programske podrške zasnovane na Android servisu

3.1 Ključne komponente sistema

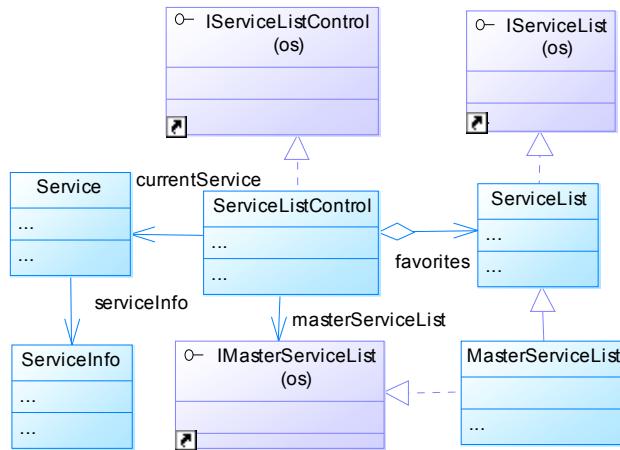
Aplikativna Java sprega, sa funkcijama značajnim za digitalnu televiziju (u daljem tekstu DTV), sastoji se od nekoliko celina, gde je najznačajnija DTVManager komponenta. Ona u sebi sadrži IDTVManager spregu, kao što ostale komponente u sebi sadrže odgovarajuće sprege (npr. ServiceManager sadrži IServiceComponent) kao što je prikazano na slici 3.2.



Slika 3.2 Komponente Android DTV servisa

Android DTVManager se sastoji od sledećih modula (kao što je prikazano na slici 3.2): **IDTVManager**, **IVideoControl**, **IAudioControl**, **IEpgControl**, **IMhegControl**, **ISubtitleControl**, **ITelletextControl**, **IServiceList**, **IConditionalAccessControl**, **IServiceListControl**, **ISetupControl**, **IReminderControl**, **IMasterServiceList** i **IPvrControl**.

Ključne komponente su: **IDTVManager**, **IServiceListControl**, **IMasterServiceList** i **IServiceList**. **IServiceListControl** (kako je i prikazano na slici 3.3) predstavlja glavnog kontrolora zaduženog za listu kanala. Sastoji se od glavne liste kanala (koju predstavlja komponenta **IMasterServiceList**) i listâ omiljenih kanala (koje su predstavljene komponentom **IServiceList**).



Slika 3.3 Struktura komponente IServicelistControl

Svaki kanal je opisan u okviru klase Service. Klasa se sastoje od informacija poput: ime kanala, tip kanala (digitalni, analogni, radio), indeks kanala u glavnoj listi kanala, informacije da li je sadržaj zaštićen (eng. Scrambled) i informacije o brojevima pojedinačnih komponenti dostupnih na kanalu (broj audio traka, postojanje teleteksta, prevoda, itd.). DTV aplikativna Java sprege je stvarana tako da se podaci ne dupliraju u sistemu. Programska podrška televizijskog prijemnika čuva sve potrebne informacije, tako da se te informacije ne dupliraju u Javi, već se po potrebi traže od programske podrške televizijskog prijemnika.

Neki od modula DTV aplikativne Java sprege koje uključuje Android servis su:

- IPvrControl - predstavlja komponentu ličnog video snimača, koji se koristi za kontrolu DTV sadržaja i njegovo skladištenje, kao i kasnije prikazivanje;
- IEpgControl - predstavlja komponentu elektronskog programskog vodiča, koja se koristi za prikazivanje dodatnih informacija o DTV sadržaju i dostupnosti emisija;
- ITeletextControl, ISubtitleControl i IMhegControl - su komponente koje se koriste za kontrolu i prikaz odgovarajućih podataka iz toka podataka;
- IReminderControl - predstavlja komponentu koja se koristi za podsetnike (događaje u kalendaru koji su povezani sa DTV sadržajem);
- ISetupControl - komponenta koja se koristi za skladištenje i preuzimanje različitih podešavanja za DTV uređaj, poput podešavanja zemlje, regionala, vremenske zone itd;
- IVideoControl i IAudioControl - vode računa o prikazivanju zvučnih i video podataka dobavljenih iz toka podataka;
- IConditionalAccessControl - vodi računa o zaštićenim kanalima i drugom zaštićenom sadržaju, poput roditeljske kontrole.

4. Programsко rešenje

U ovom poglavlju je dat detaljan opis uvedenog Android servisa, prikaz sprege Java i C koda, kao i način korišćenja servisa u Android aplikaciji.

4.1 Tipovi Android servisa i njihove specifičnosti

Da bi mogli pristupiti servisu koji predstavlja drugi proces/aplikaciju u odnosu na vašu aplikaciju, potrebno je koristiti mehanizme međuprocesne komunikacije (eng. Inter-process communication - IPC), jer u Android OS korisnik ne može da pristupi memorijском prostoru drugog procesa. Na primer, da bi Android aplikacija mogla da upravlja servisom za telefoniranje potrebno je koristiti IPC pozive.

Postoje dva tipa Android servisa [5]:

- Servisi koji se pokreću iz neke aplikacije pozivom metode `startService()`. To su obično servisi koji se pokreću u pozadini da obavljaju jedan posao neodređeno vreme i obično ne vraćaju informaciju o izvršenju operacije korisniku koji ga je pokrenuo;
- Servisi na koje se korisnici povezuju korišćenjem metode `bindService()`.

Postoji više mehanizama koji se koriste za udaljene servise, ali u ovom radu predmet našeg interesovanja je AIDL, tako da ćemo njega i obraditi. AIDL datoteka predstavlja spregu u kojoj su sadržane metode kojima želimo da pristupimo iz klijentske aplikacije. Potpuno identične AIDL datoteke moraju biti i na strani poslužioca i na strani klijenta u okviru istih paketa. Sintaksa AIDL datoteke je veoma slična Javinoj, podržani su svi Java primitivni tipovi, ali i svi tipovi koji u sebi sadrže parselabilan protokol. Rezultat prevođenja AIDL datoteke je mašinski generisana klasa koju ne treba menjati.

4.2 Povezivanje i raskidanje veze sa Android servisom

Klijenti se po potrebi vezuju na udaljeni servis, pozivaju metode servisa i nakon završenog posla prekidaju vezu sa servisom. Servisi koji rade po principu povezivanja mogu biti istovremeno dostupni većem broju korisnika, a kada svi korisnici prekinu vezu sa servisom, on se gasi, da ne bi zauzimao sistemske resurse. Da bismo u potpunosti iskoristili mogućnost Androida, servis koji je tema ovog rada smo organizovali tako da se može i pokrenuti, ali da može i prihvati klijente koji se na njega povezuju. To je urađeno tako što su dodate dve povratne metode koje se koriste u Androidu: `onStartCommand()` (koja omogućava da se izvrše potrebne operacije za pokretanje servisa) i metoda `onBind()` (koja dozvoljava povezivanje klijenata na servis). Ključna metoda u servisu je baš metoda `onBind()` koja kao povratnu vrednost vraća objekat IBinder klase, koji se koristi za komunikaciju. U ovoj metodi se stvara klasa naslednica klase IDTVManager.Stub i realizuju se sve metode navedene u AIDL datoteci. Klasa IDTVManager.Stub je mašinski stvorena klasa (nastala od AIDL datoteke) i nju nasleđujemo da bismo realizovali sve metode navedene u AIDL datoteci. Prilikom realezacije metoda iz AIDL datoteke potrebno je voditi računa da se posao, ako je dugotrajan, izmesti iz glavne niti u kojoj se aplikacija izvršava u zasebnu nit.

Povezivanje klijentske aplikacije sa servisom se obavlja tako što se poziva metoda `bindService(intent, connection, flag)` koja će vezati klijentsku aplikaciju za servis. Prvi parametar je objekat klase *Intent* kojim se identificuje servis, drugi parametar je klasa koja predstavlja vezu sa servisom i koja će obezrediti pozivanje metoda servisa, a treći parametar definiše da se servis stvara ovom komandom, ako pre toga nije postojao. Za komunikaciju klijenta sa servisom koristi se veza (drugi parametar metode `bindService(intent, connection, flag)`), realizovana kao klasa koja nasleđuje ServiceConnection klasu. Ova klasa ima dve povratne metode: `onServiceConnected()`, koja se poziva kada se uspostavi komunikacija sa servisom i metoda `onServiceDisconnected()`, koja se poziva kada se veza sa servisom raskine. Metoda `onServiceConnected()` se koristi da bi se inicijalizovala klasa koja poseduje metode servisa. Ova klasa realizuje IDTVManager spregu (mašinski stvorenu) koja sadrži sve metode koje servis ima i omogućava da se njihovim pozivom zapravo pozovu metode u servisu.

Raskidanje veze sa servisom se obavlja pomoću metode `unbindService(connection)`. Ovaj poziv će izazvati poziv metode `onServiceDisconnected()` klase ServiceConnection na klijentskoj strani, kojim se klijentu sugerise da je komunikacija sa servisom završena.

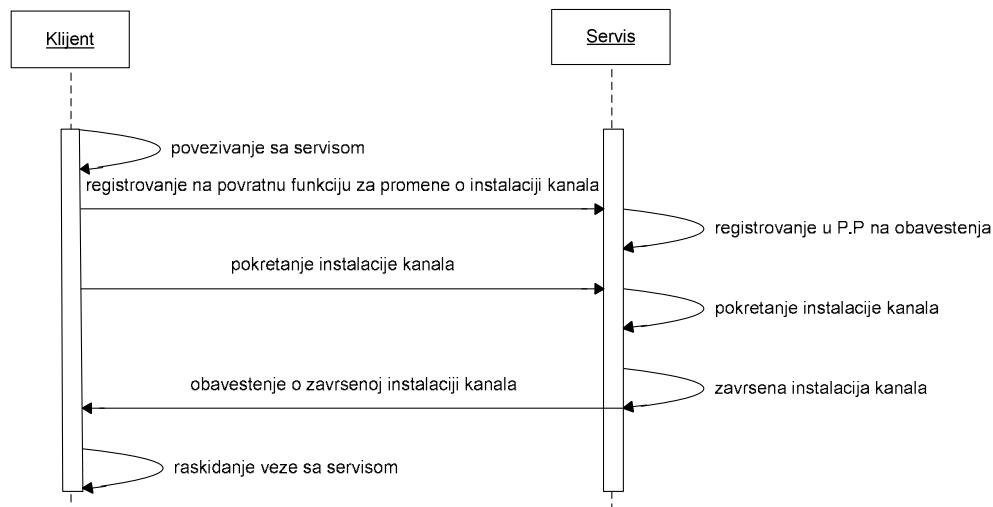
4.3 Obaveštenja i povratne metode iz Android servisa

Na strani servisa postoje realizovane metode koje kada dobiju određene asinhronne informacije od programske podrške televizijskog prijemnika, iste mogu proslediti do klijenata,

pod uslovom da su se klijenti prethodno registrovali za dobijanje tih informacija. Informacije koje klijentska strana dobija mogu biti:

- desila se promena događaja koji se trenutno emituje;
- desila se promena informacija o elektronskom programskom vodiču;
- napredak pri postupku pretraživanja kanala;
- promenjen je broj trenutno pronađenih radio kanala prilikom pretraživanja kanala;
- promenjen je broj trenutno pronađenih televizijskih kanala prilikom pretraživanja kanala;
- pronađen je novi radio kanal sa određenim nazivom;
- pronađen je novi televizijski kanal sa određenim nazivom;
- završeno je pretraživanje kanala;
- ubačen/izbačen je modul za uslovni pristup (eng. Conditional access modul - CAM)
- obaveštenje da je potrebno osvežiti određene grafičke komponente (tekst, prevod).

Klijent će po uspostavljanju veze sa servisom proslediti pokazivač na svoj kod koji će servis pozivati kada dobije obaveštenje od programske podrške televizijskog prijemnika o određenom događaju. Najprikladnije mesto za prosleđivanje pokazivača klijentskog koda servisu je metoda `onServiceConnected()`, jer je tu sigurno veza sa servisom uspostavljena. Primer realizacije obaveštenja je dat na slici 4.1:



Slika 4.1 Implementacija sistema obaveštenja

4.4 Prenos složenih klasa između različitih procesa

AIDL mehanizam podržava razmenu podataka među povezanim stranama, ali sve dok su ti podaci primitivnog tipa, poput tipova *int*, *boolean*, *char* i ugrađenih Javinih tipova poput *String*, *List*, *CharSequence* i slično. Da bi se moglo prenositi složene klase poput klasa *Service.java*, *EpgEvent.java*, *RecordedEvent.java*, potrebno je da svaka od njih u sebe uključuje i parselabilan protokol. Kada se koristi AIDL mehanizam za prenos podataka između procesa obavlja se skladištenje i prenos objekta (eng. Marshalling) tokom procedure slanja objekta kao parametra funkcije, a suprotan proces prilikom prijema objekta kao parametra funkcije (eng. unmarshalling). Kada zelimo da prenesemo složene objekte ova procedura nije moguća, pa zato sve složene klase moraju da sadrže parselabilan protokol, koji se koristi za rastavljanje složenih klasa na primitivne tipove i predstavlja osnovu za međuprocesnu komunikaciju u Androidu. Svaka klasa pored Java realizacije, mora da ima i istoimenu AIDL datoteku u kojoj je data puna putanja do klase i gde je naglašeno da klasa uključuje parselabilan protokol. U Java realizaciji je potrebno uključiti nekoliko dodatnih metoda, od kojih su najznačajnije `writeToParcel(Parcel dest, int flags)` i `readFromParcel(Parcel in)`. U ovim metodama se sva polja složene klase iščitavaju iz objekta tipa *Parcel*, odnosno zapisuju u objekat tipa *Parcel*, u zavisnosti od toga da li je u pitanju klijent ili servis.

4.5 Moduli i metode Android servisa

Android servis sadrži DTV Java spregu koja se sastoji od sledećih paketa:

- com.android.server – sadrži Android servis koji je realizovan kao *DTVManagerService*, a moguće mu je pristupiti preko metode `bindService()`;
- android.dtv.audio – sadrži klase neophodne za upravljanje zvukom televizijske aplikacije;
- android.dtv.audio.exception – sadrži izuzetke pri upravljanju zvukom;
- android.dtv.ca – sadrži klase za upravljanje CAM modulom televizijske aplikacije;
- android.dtv.epg – sadrži klase za upravljanje elektronskim programskim vodičem televizijske aplikacije;
- android.dtv.mheg – sadrži klase za upravljanje grafičkim interaktivnim sadržajem televizijske aplikacije;
- android.dtv.pvr – sadrži klase za upravljanje ličnim video snimačem televizijske aplikacije;
- android.dtv.reminder – sadrži klase za upravljanje podsetnicima u televizijskoj aplikaciji;

- android.dtv.service – sadrži klase za upravljanje listama servisa televizijske aplikacije;
- android.dtv.setup – sadrži klase za podešavanja televizijske aplikacije;
- android.dtv.subtitle – sadrži klase za upravljanje prevodom u televizijskoj aplikaciji;
- android.dtv.teletext – sadrži klase za upravljanje teletekstom u televizijskoj aplikaciji;
- android.dtv.utils – sadrži uslužne klase televizijske aplikacije;
- android.dtv.video – sadrži klase neophodne za upravljanje video sadržajem televizijske aplikacije;
- android.dtv.video.exception – sadrži izuzetke pri upravljanju video sadržajem;
- android.os – sadrži sve sprega koje koriste klijentske aplikacije.

4.5.1 Paket android.os

Ovaj paket sadrži sve module koji su neophodni za normalno funkcionisanje jedne DTV aplikacije. Moduli su predstavljeni kao AIDL datoteke, pa je potrebno da identičan sadržaj bude na strani servisa i na strani klijenata. Uključuje sledeće module u sebe:

- IDTVManager - glavna sprega servisa koja se dobija kao rezultat metode `bindService();`
- IAudioControl - sprega za upravljanje zvukom;
- IConditionalAccessControl - sprega za upravljanje CAM modulom;
- IEpgControl - sprega za upravljanje elektronskim programskim vodičem;
- IMasterServiceList - sprega za osnovnu listu kanala;
- IMhegControl - sprega za upravljanje grafičkim sadržajem interaktivne televizije;
- IPvrControl - sprega za lični video snimač;
- IReminderControl - sprega za podsetnike;
- IServiceList - sprega za liste servisa;
- IServiceListControl - sprega za upravljanje listama servisa;
- ISetupControl - sprega za podešavanja;
- ISubtitleControl - sprega za upravljanje prevodom;
- ITTeletextControl - sprega za upravljanje teletekstom.

4.5.2 Paket com.android.server

Ovaj paket sadrži modul koji predstavlja skup svih sprega koje klijentska strana može koristiti, objedinjenih u klasi *DTVManagerService*.

4.5.3 Paket android.dtv.pvr

Ovaj paket sadrži klase neophodne za kontrolu ličnog video snimača televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `getRecordedEventListCount()` – kao rezultat daje broj snimljenih događaja;
- `getRecordedEvent(int index)` – kao rezultat daje informacije o snimljenom događaju za dati indeks;
- `startTimeShift(Service service)` – pokreće odgođeno snimanje za dati servis;
- `startPlayback(RecordedEvent event)` – pokreće prikazivanje snimljenog događaja.

4.5.4 Paket android.dtv.reminder

Ovaj paket sadrži klase neophodne za upravljanje podsetnicima vezanih za događaje televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `addReminder(ReminderItem item)` – dodaje podsetnik za dati događaj;
- `deleteReminder(ReminderItem item)` – briše podsetnik za dati događaj;
- `getReminderItem(int index)` – kao rezultat daje informacije o podsetniku za dati indeks.

4.5.5 Paket android.dtv.service

Ovaj paket sadrži klase neophodne za upravljanje listama dostupnih servisa DTV aplikacije. Ovde postoje dve glavne liste: glavna lista kanala i liste omiljenih kanala. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `fullScan()` – potpuno pretraživanje kanala;
- `getService(int index)` – kao rezultat daje detalje o servisu za dati indeks;
- `getServiceListCount()` – kao rezultat daje informaciju o broju dostupnih servisa;
- `getCurrentService()` – kao rezultat daje informacije o servisu koji se trenutno emituje;
- `setCurrentService(int index)` – izdaje komandu da se servis sa unetim indeksom postavi za onaj koji se trenutno emituje.

Informacije koje su dostupne o svakom servisu su sledeće:

- `getFreq()` – daje informaciju o frekvenciji na kojoj se emituje servis;
- `getType()` – daje informaciju o tipu servisa (može biti digitalni, analogni, radio);
- `getName()` – daje informaciju o imenu servisa;
- `isScrambled()` – daje informaciju o tome da li je servis zaštićen;
- `getIndex()` – daje informaciju o indeksu servisa u glavnoj listi servisa;

4.5.6 Paket android.dtv.setup

Ovaj paket sadrži klase neophodne za upravljanje podešavanjima televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `getCountry()` – kao rezultat vraća informaciju o trenutnoj zemlji podešenoj u programskoj podršci televizijskog prijemnika;
- `getTimeZone()` – kao rezultat vraća informaciju o trenutnoj vremenskoj zoni;
- `setCountry(int countryCode)` – podešava trenutnu zemlju na osnovu unetog koda;
- `getTimeDate()` – kao rezultat vraća informaciju o trenutnom vremenu iz toka podataka.

4.5.7 Paket android.dtv.audio

Ovaj paket sadrži klase neophodne za upravljanje zvukom uz pomoć programske podrške televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `getAudioTrack(int index)` – kao rezultat vraća zvučni tok za dati indeks;
- `getAudioTrackCount()` – kao rezultat vraća broj dostupnih zvučnih tokova za trenutno aktivini servis;
- `getCurrentVolume()` – kao rezultat vraća trenutnu jačinu zvučnog toka;
- `setCurrentAudioTrack(int index)` – postavlja zvučni tok sa datim indeksom kao aktivan;
- `setCurrentVolume(int currentVolume)` – postavlja jačinu zvučnog toka.

4.5.8 Paket android.dtv.epg

Ovaj paket sadrži klase neophodne za upravljanje elektronskim programskim vodičem uz pomoć programske podrške televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `getEpgEventListCount()` – kao rezultat vraća broj dostupnih događaja;
- `getEpgEvent(int index)` – kao rezultat vraća događaj za dati indeks.

Svaki od događaja pruža sledeće informacije:

- `getDescription()` – kao rezultat vraća opis događaja;
- `getDuration()` – kao rezultat vraća trajanje događaja;
- `getEndTime()` – kao rezultat vraća vreme završetka događaja;
- `getExtendedDescription()` – kao rezultat vraća detaljniji opis događaja;
- `getName()` – kao rezultat vraća ime događaja;
- `getStartTime()` – kao rezultat vraća vreme početka događaja.

4.5.9 Paketi android.dtv.mheg, android.dtv.subtitle, android.dtv.teletext

Skup ovih paketa se bavi predstavljanjem grafike dobijene od programske podrške televizijskog prijemnika, pa će svi biti opisani zajedno. Između ostalih, u ovim modulima se izdvajaju sledeće metode:

- `hide()` – sklanja dati element sa Android grafičke ravni;
- `show(int alpha)` – prikazuje dati element na Android grafičkoj ravni sa određenim nivoom providnosti *alpha*;
- `sendInputControl(int ctrl, int param)` – šalje određenu korisničku kontrolu programskoj podršci televizijskog prijemnika;
- `scale(int x, int y)` – menja veličinu grafičkog elementa na određene parametre.

4.5.10 Paket android.dtv.video

Ovaj paket sadrži klase neophodne za upravljanje videom uz pomoć programske podrške televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `initPlayer(int serviceIndex)` – pokreće reprodukciju servisa sa datim indeksom;
- `deinitPlayer()` – zaustavlja reprodukciju aktivnog servisa;
- `getBrightness()` – kao rezultat vraća nivo osvetljenja;
- `setBrightness(double brightness)` – podešavanje osvetljenja na određeni nivo;
- `setScaling(int width, int height, int x, int y)` – promena veličine emitovanog videa na date ulazne parametre;
- `setContrast(double contrast)` – podešavanje kontrasta na dati ulazni nivo;
- `getContrast()` – kao rezultat vraća trenutno podešeni kontrast.

4.5.11 Paket android.dtv.ca

Ovaj paket sadrži klase neophodne za upravljanje zaštićenim sadržajem televizijskog prijemnika. Između ostalih, u ovom modulu se izdvajaju sledeće metode:

- `start()` – pokreće korišćenje CAM modula;
- `sendInputControl(int ctrl, int param)` – slanje odgovarajuće korisničke komande;
- `stop()` – prestanak korišćenja CAM modula.

Zaključno sa ovim paketom oformljen je skup DTV Java modula uz čiju pomoć se može odvijati nesmetana komunikacija i razmena podataka sa programskom podrškom televizijskog prijemnika. Sledeći korak je spajanje DTV Java modula koji su napisani u Java programskom jeziku i programske podrške televizijskog prijemnika, koja je pisana u C kodu. Spregu između

ova dva sloja programske podrške predstavlja JNI [6]. Osnovne mane JNI realizacije o kojima je potrebno dodatno povesti računa su:

- greške koje mogu destabilizovati celu virtuelnu mašinu;
- gubi se platformska nezavisnost koda;
- ne radi Javin sistem oslobođanja nekorišćene memorije (eng. garbage collection)

Sa druge strane, pogodnosti su te da imamo direktni pristup fizičkoj arhitekturi i da se JNI kod najčešće brže izvršava u odnosu na Java kod.

Naš JNI sloj programske podrške se sastoji od deklaracija metoda koje su prethodno opisane u Java paketima i njihove realizacije u C kodu. Deklaracije metoda se nalaze u okviru Java paketa, dok se sama realizacija nalazi u zasebnim C modulima koji su Javi dostupni u obliku deljene biblioteke. Ovde je realizovano dvosmerno prozivanje metoda:

- smer od Java aplikacije ka programsкој podršci televizijskog prijemnika;
- smer od programske podrške televizijskog prijemnika ka Java aplikaciji.

Prvi smer prozivanja je više nego pravolinijiški: Java aplikacija proziva Java servis, koji preko DTV Java sprege spušta poziv do JNI sloja. JNI sloj je taj koji zatim proziva odgovarajući metodu programske podrške televizijskog prijemnika, ali ne direktno, već preko izloženih modula apstrakcije programske podrške televizijskog prijemnika. Na taj način je omogućeno da se sloj programske podrške televizijskog prijemnika menja, a da to ne mora da iziskuje promene u slojevima iznad njega.

Drugi smer prozivanja je nešto složeniji. Kada se određeni DTV događaj desi, programska podrška televizijskog prijemnika proziva povratnu JNI metodu (koju smo prethodno registrovali tako što smo prosledili pokazivač na tu JNI metodu), a ta metoda zatim proziva odgovarajući DTV Java metodu. Ta metoda preko servisa obaveštava sve klijente koji su se povezali da prate promene nekog DTV događaja da se promena desila.

5. Ispitivanje i verifikacija

U okviru ovog rada obavljena su ispitivanja realizovanih modula na dva nivoa. Na prvom nivou, korišćenjem nezavisnih JUnit ispitnih slučajeva, verifikovani su svi moduli koji su predmet ovog rada. Drugi nivo ispitivanja predstavlja zasebna Android aplikacija. Svi realizovani moduli su uključeni u prethodno pomenutu Android aplikaciju i utvrđeno je da izolovani ispitni slučajevi ne pružaju validne rezultate, tj. metode koje vraćaju ispravnu vrednost kada se izolovano pozivaju, vraćaju neispravne rezultate kada se uključe u televizijsku aplikaciju, tako da je kod u iteracijama ispravljan da bismo došli do potpuno ispravnog rešenja.

Grupe funkcionalosti koje su ispitane u ovim slučajevima su:

- ručno pretraživanje kanala;
- prebacivanje kanala;
- prikaz informacija o kanalu pozivanjem povratne metode;
- promena više zvučnih tokova na jednom kanalu;
- prikaz informacija elektronskog programskog vodiča

U okviru svake od grupe testova, verifikovano je više funkcionalnosti korišćenog rešenja.

5.1 Ručno pretraživanje kanala

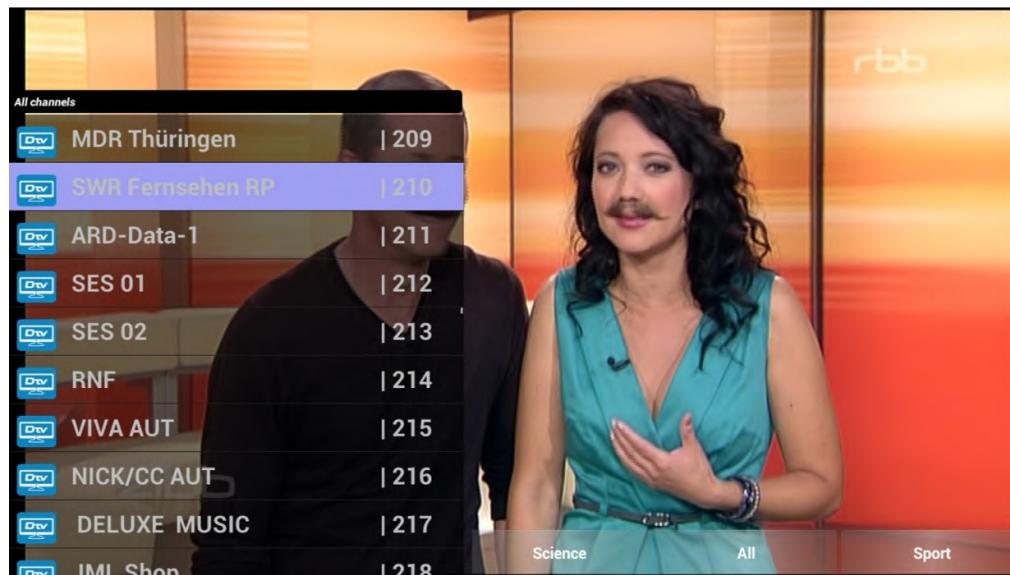


Slika 5.1 Ručno pretraživanje kanala

Ispitivanje ove funkcionalnosti je obavljeno tako što su se kao ulazni parametri metode postavljali parametri sa portala KingOfSat [7], a kao rezultat su uvek pronađeni svi kanali na unesenim frekvencijama. Time se potvrđuje sledeće:

- uspešno uspostavljanje veze klijentske Android aplikacije sa Android servisom;
- uspešno pozivanje metode servisa za ručnu pretragu kanala;
- uspešno prosleđivanje složenog objekta sačinjenog od parametara za skeniranje od klijentske strane, preko servisa, do programske podrške televizijskog prijemnika;
- pronalaženje svih očekivanih kanala sa vrednostima unesnih parametara, što predstavlja ispravan odgovor programske podrške televizijskog prijemnika Android servisu;
- obaveštavanje klijenta da je proces pretrage uspešno završen i puštanje prvog pronađenog kanala.

5.2 Prebacivanje kanala



Slika 5.2 Prebacivanje kanala

Ispitivanje ove funkcionalnosti je potvrđeno pozivanjem metoda za popunjavanje liste dostupnih kanala i uspešno prebacivanje sa trenutnog kanala na kanal sa indeksom 210 u listi kanala. Time se potvrđuje sledeće:

- uspešno pokretanje reprodukcije prvog servisa iz liste;
- ispravno popunjavanje liste svih dostupnih kanala;
- ispravna vrednost informacije o kanalu da li je isti zaštićen (kanal je slobodan, pa je moguće pratiti njegovu sliku i zvuk, inače bi bile dostupne samo informacije o kanalu);
- ispravno prebacivanje kanala po indeksu, jer je uspešno prebačeno na kanal sa indeksom 210.

5.3 Prikaz informacija o kanalu pozivom povratne metode



Slika 5.3 Prikaz informacija o trenutnom kanalu

Ispitivanjem ove funkcionalnosti je potvrđeno funkcionisanje razmene složenih struktura između programske podrške televizijskog prijemnika, Android servisa i Android klijentske aplikacije. Takođe je potvrđeno ispravno funkcionisanje povratnih metoda od programske podrške televizijskog prijemnika do Android servisa, koji zatim obaveštava klijentske aplikacije o promeni. Time se potvrđuje sledeće:

- prilikom menjanja kanala, javlja se povratna metoda da su dostupne nove informacije o trenutno gledanom kanalu;
- uspešno su preuzete nove informacije o kanalu: naziv kanala, vreme početka emitovanja tekućeg događaja, kraj emitovanja trenutnog događaja, nazivi tekućeg i sledećeg događaja, trenutno vreme dostupno iz toka podataka.

5.4 Promena više zvučnih tokova na jednom kanalu

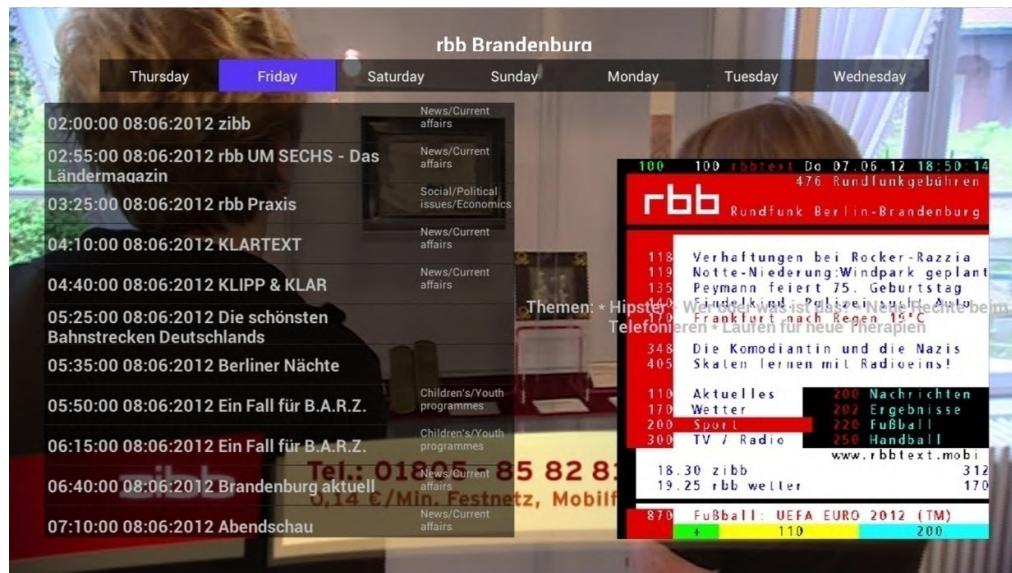


Slika 5.4 Prikaz liste dostupnih zvučnih tokova na trenutnom kanalu

Ispitivanjem ove funkcionalnosti potvrđena je ispravnost modula za kontrolu zvuka. Time se utvrđuje sledeće:

- uspešno su prikupljeni podaci o dostupnim zvučnim tokovima na trenutno gledanom kanalu;
- podaci su uspešno poslati do klijentske aplikacije;
- klijentska aplikacija je prozvala metodu za promenu aktivnog zvučnog toka i programska podrška televizijskog prijemnika je promenu izvršila.

5.5 Elektronski programski vodič



Slika 5.5 Prikaz elektronskog programskega vodiča

Ispitivanjem ove funkcionalnosti potvrđena je ispravna realizacija modula za kontrolu elektronskog programskega vodiča. Programskega vodiča prikazuje informacije o događajima za trenutni kanal u narednih 7 dana. Time se utvrđuje sledeće:

- ispravno je podešen vremenski odsečak u kojem se vrši sakupljanje informacija o kanalu;
- ispravno je izvršen izbor informacija za prikaz (u ovom slučaju je obeleženo prikazivanje svih događaja za petak);
- informacije koje su dobijene od programske podrške televizijskog prijemnika su u ispravnom formatu (vreme prikaza se podudara sa programskom šemom datog kanala, naziv emisije se podudara sa programskom šemom datog kanala, tip emisije se podudara sa tipom pronađenim na programskoj šemi za dati kanal).

6. Zaključak

U ovom radu je opisano jedno rešenje Android DTV servisa i sprege navedenog servisa sa programskom podrškom televizijskog prijemnika proizvođača iWedia. Rešenje je realizovano na Marvell BG2 SOC platformi sa satelitskim biračem kanala. Da bi nesmetano više Android aplikacija (koje predstavljaju klijente u odnosu na programsku podršku televizijskog prijemnika) moglo da koristi programsku podršku televizijskog prijemnika, bilo je potrebno postojeće rešenje DTV Java sprege unaprediti tako da podržava više korisnika. Da bi se omogućila osnovna funkcionalnost programske podrške televizijskog prijemnika svakoj Android aplikaciji, realizovan je DTVManager koji sadrži sledeće komponente:

- IDTVManager - glavna sprega servisa;
- IAudioControl - sprega za upravljanje zvukom;
- IConditionalAccessControl - sprega za upravljanje CAM modulom;
- IEpgControl - sprega za upravljanje elektronskim programskim vodičem;
- IMasterServiceList - sprega za osnovnu listu kanala;
- IMhegControl - sprega za upravljanje grafičkim sadržajem interaktivne televizije;
- IPvrControl - sprega za lični video snimač;
- IReminderControl - sprega za podsetnike;
- IServiceList - sprega za liste servisa;
- IServiceListControl - sprega za upravljanje listama servisa;
- ISetupControl - sprega za podešavanja;
- ISubtitleControl - sprega za upravljanje prevodom;
- ITeletextControl - sprega za upravljanje teletekstom.

Rešenje je ispitivano na ciljnoj fizičkoj arhitekturi sa Comedia programskom podrškom televizijskog prijemnika i rezultati su upoređivani sa ispitnom konzolnom aplikacijom.

Upoređivanjem ponašanja aplikacije sa rezultatima koje metode vraćaju prilikom JUnit ispitivanja utvrđeni su identični rezultati na taj način su potvrđene sledeće grupe ispitnih slučajeva:

- ručno pretraživanje kanala;
- prebacivanje kanala;
- prikaz informacija o kanalu pozivanjem povratne metode;
- promena više zvučnih tokova na jednom kanalu;
- prikaz informacija elektronskog programske vodiča

Dato rešenje će sa minimalnim izmenama na strani Android servisa raditi i sa drugim biračima kanala, na primer sa zemaljskim biračem kanala. Takođe, predloženo rešenje pošto je pisano u Java programskom jeziku je nezavisno od konkretne platforme, pa je uz promenu programske podrške televizijskog prijemnika, bez promene servisa, moguće nastaviti korišćenje istog Android servisa [8].

7. Literatura

- [1] iWedia sajt, www.iwedia.com, učitano 18.06.2012.
- [2] Vladimir Kovačević, Miroslav Popović: Sistemska programska podrška u realnom vremenu, Univerzitet u Novom Sadu, Fakultet Tehničkih Nauka, 2002
- [3] Sajt Android podrške za razvoj, www.developer.android.com, učitano 18.06.2012.
- [4] JNI Oracle sajt, www.oracle.com/javase/1.4.2/docs/guide/jni/spec/functions.html, učitano 18.06.2012.
- [5] Predavanja iz predmeta Projektovanje namenskih računarskih struktura 1, <http://www.rt-rk.uns.ac.rs/studijski-program-2009/vi-2009/pnrs1/737-pnrs1-predavanja>
- [6] Sajt JNI podrške, <http://developer.android.com/sdk/ndk/overview.html>.
- [7] Sajt KingOfSat, <http://en.kingofsat.net/>
- [8] M.Vidakovic, N.Teslic, T.Maruna, and V.Mihic: *Android4TV: a proposition for integration of DTV in Android devices*, IEEE 30th International Conference on Consumer Electronics (ICCE), Las Vegas, January 2012, pp. 441-442