UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Jurij Sirše

Razvoj aplikacije Baliranje trave za mobilno platformo Android

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Saša Divjak

Ljubljana, 2012

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00140/2011
Datum: 01.09.2011

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **JURIJ SIRŠE**

Naslov: **RAZVOJ APLIKACIJE "BALIRANJE TRAVE" ZA MOBILNO PLATFORMO ANDROID**

DEVELOPMENT OF THE APPLICATION "BALING GRASS" FOR THE MOBILE PLATFORM ANDROID

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Predstavite postopek in probleme pri opravljanju storitve baliranja trave oziroma slame. Predlagajte računalniško aplikacijo, ki bi te probleme reševala s pomočjo pametnega telefona in mobilnega operacijskega sistema Android. Predstavite razvojno okolje Eclipse IDE in vtičnik Android SDK. Opišite različico Jave, ki se uporablja za razvoj aplikacij Android in SQLite za delo z podatkovno bazo.
Razvijte in opišite aplikacijo Baliranje. Predstavite podatkovni model, entitete in ekstraktorje. Podajte poslovno logiko in komponente Androida, ki tečejo v ozadju. Opišite uporabniški vmesnik in xml datoteke AndroidManifest, brez katere se aplikacija sploh ne požene.

Mentor:

prof. dr. Saša Divjak

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani **Jurij Sirše**,

z vpisno številko **63060368**,

sem avtor diplomskega dela z naslovom:

Razvoj aplikacije Baliranje za mobilno platformo Android

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Saše Divjak
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 14.3.2012 Podpis avtorja: _____

ZAHVALA

Sprva se iskreno zahvaljujem prof. dr. Saši Divjak za pomoč, jasna navodila in strokovno svetovanje pri izdelavi diplomske naloge.

Največjo zahvalo si zaslužita starša Marija in Milan, ki sta mi omogočila študij na Fakulteti za računalništvo in informatiko v Ljubljani in me sproti spodbujala ter nudila vso podporo, da sem dosegel še en življenjski cilj.

Ob tej priložnosti se moram zahvaliti tudi podjetju Agito d.o.o., ki me je tekom našega sodelovanja izšolalo v boljšega in bolj izkušenega razvijalca programske opreme.

KAZALO

SEZNAM UPORABLJENIH KRATIC IN OKRAJŠAV

IDE – Integrated Development Environment

SDK – Software Development Kit

SQL – Structured Querry Language

SMS – Short Message Service

XML – Extensible Markup Language

OHA – Open Handset Alliance

2D – Two Dimensional

3D – Three Dimensional

SGL – Scalable Games Language

RIM – Research In Motion

HTC – High Tech Computer

iOS – Operating Sistem (črka i je pripona, ki jo uporablja Apple pri svojih produktih)

OS – Operating System

ME – Micro Edition

Inc – Incorporation

SSL – Secure Socket Layer

OSGi – Open Services Gateway initiative

JDT – Java Development Tools

ADT – Android Development Tools

JRE – Java Runtime Environment

FCFS – First-come, first-served

SD – Secure Digital

POVZETEK

V uvodu diplomskega dela je opisan namen in uporabnost razvite aplikacije. Temu sledi kratka predstavitev baliranja trave oziroma slame, in težav, na katere smo naleteli pri opravljanju storitve baliranja ter predlog aplikacije, ki bi te težave reševala.

Sledi splošni opis naprave pametni telefon in mobilnega operacijskega sistema Android, kateremu priljubljenost med uporabniki vse bolj narašča. Nato sta predstavljena razvojno okolje Eclipse IDE in plug-in Android SDK. Nekaj besed je namenjenih tudi programskemu jeziku Java, ki se uporablja za razvoj aplikacij Android in SQLite za delo s podatkovno bazo.

Osrednji del diplomskega dela je namenjen opisu in razlagi aplikacije Baliranje trave. V tem poglavju je najprej predstavljen podatkovni modul z zbirko podatkov, entitetami in ekstraktorji. Sledi še podpoglavje s poslovno logiko in opisom komponent Androida, ki tečejo v ozadju. Prav tako sta opisana tudi uporabniški vmesnik in xml datoteka AndroidManifest. Nekaj besed je namenjenih tudi testiranju omenjene aplikacije, predstavljene so tudi nekatere ideje, ki bi bile vključene v razvoj naslednje različice aplikacije.

Ključne besede:

baliranje, mobilni operacijski sistem Android, pametni telefon, razvoj, uporabniški vmesnik

ABSTRACT

The introduction of the thesis describes the purpose and usefulness of the developed application. This is followed by a presentation of baling grass or straw and the problems we encountered when we were performing the service of baling hay and a proposal for a application, which would solve these problems.

Next is a general description of the smart phone and mobile operating system Android, which recently popularity is growing. Then followed by a presentation of the development environment Eclipse IDE and the Android SDK. Then some words are intended for the Java programming language used to develop Android applications and SQLite for working whit the database.

The main part of the thesis is intended for the explanation of the application Baling grass, where first of all is a presentation of the data model, entities and extractors. Following chapters focus on the business logic and a description of Android components that run in the background. Next in line is a description of the user interface and the AndroidMainfest xml file. A few words are intended for the testing of this application and also a brief presentation of some ideas that would be included in the development of the next version of the application.

Key words

baling, mobile operating system Android, smart phone, development, user interface

1 UVOD

V diplomskem delu je predstavljena mobilna aplikacija Baliranje trave, ki omogoča traktoristom (oziroma izvajalcem storitve baliranja) enostavno vodenje evidence storitve (določanje števila in tipa narejenih bal), vpogled v seznam strank, ki niso poravnale stroške baliranja, kmetom pa enostavno naročanje storitve s telefonskim klicem ali preko sporočila SMS.

Idejo za to aplikacijo sem dobil v času počitnic pred začetkom študija na Fakulteti za računalništvo in informatiko, saj sem takrat pomagal kot traktorist, ki je ovijal bale. Takrat še nisem imel ustreznega znanja, da bi aplikacijo lahko razvil, prav tako ni bilo naprodaj cenovno ugodne naprave, ki bi takšno aplikacijo poganjala. V zadnjih letih smo priča izjemnemu tehnološkemu napredku pametnih telefonov, ki postajajo vse manjši, zmogljivejši ter cenovno ugodnejši. Najbolj znana imena na trgu pametnih telefonov so HTC, Samsung, Apple, RIM itd. Te naprave lahko poganjajo različne mobilne operacijske sisteme kot so Symbian, iOS, Blackberry OS, Windows Mobile Phone 7 in vse bolj priljubljen Android, za katerega je aplikacija tudi razvita.

1.1 Baliranje trave

Najprimernejši čas za siliranje oziroma baliranje ovenele trave (300g - 400g suhe snovi na kilogram trave) je takoj po aprilu, ko se vreme izboljša. Baliranje omogoča zelo zgodnjo košnjo sena (prva košnja trave), hitro spravilo in visok pridelek na hektar površine. S tem načinom spravila se doseže, da v krmi po spravilu, zaradi okolja v katerem je skladiščena, začnejo potekati anaerobni biokemični procesi, ki krmi dodajo prehransko vrednost [1]. Uspešno siliranje v tem času je odločilnega pomena za uspešno gospodarjenje na kmetijah, saj v zimskem času predstavlja travna silaža glavni vir beljakovin v prehrani goveda. Lahko se silira tudi otava in otavič (druga in tretja košnja), vendar je pridelek manjši. Splošno gledano predstavlja optimalni čas za košnjo do stadija začetka latenja trav, to je čas pred katerim rastlina doseže maksimalno rast v višino in začne nakopičena hranila seliti v rezervne organe ter kasneje v seme. Siliranje prestare trave je tako ena največjih napak, ki jih kmet lahko naredi [2].

Poznamo več načinov spravila suhe trave in slame. Prva dva način spravila predstavljata spravilo v okrogle oziroma valjaste bale. Omenjena načina spravila sta primerna le, če je seno dovolj osušeno, saj je s tem povezana tudi kvaliteta shranjene krme. Pri prvem načinu izdelane bale ovijemo z več sloji folije, s čimer povzročimo nastanek biokemičnih reakcij. Pri drugem načinu spravila sena se bale izdelujejo z isto mehanizacijo kot za siliranje v okrogle bale, le da ovijanje

na koncu ni potrebno. Tretji način spravila predstavlja spravilo v obliki kvadratnih bal. Današnja mehanizacija omogoča izdelavo različnih velikosti kvadratnih bal [1], [2].

Če se kmet odloči za tak način spravila krme, mora biti pozoren na optimalno vsebnost vlage v senu ali slami. Prevelika količina vlage lahko namreč povzroči pregrevanje bale, premajhna količina vlage pa povzroči drobljenje sena in s tem manjši volumen sena na hektar [1].

1.2 Predstavitev problema

Pred leti sem ovijal bale za podjetje Kmetijska zadruga Vransko z.o.o., ki na svojem območju ponuja storitev baliranja trave in slame. To storitev v večini primerov opravljata dva traktorista, pri čemer eden z balirko balira travo ali slamo, drugi pa ovija narejene bale. Med izvajanjem te storitve sem se soočil z nekaterimi težavami, ki so povzročale nevšečnosti pri delu in so opisane v nadaljevanju.

Na začetku opravljanja počitniškega dela sem se soočil s težavo organizacije in delovanja, saj sem bil pri delu novinec. Poleg tega pa tudi ne stanujem v isti občini, kjer se je delo opravljalo, tako da nisem poznal naročnikov storitev (kmetov), niti lokacije njihovih zemljišč.

Vsak mesec je bilo potrebno izpolniti evidenčni list, v katerega se je vpisovalo koriščenje storitve baliranja. V vmesnem času sem si zapisoval potrebne podatke (datum baliranja, ime in priimek kmeta, število narejenih bal in morebitno plačilo ovijanja bal) na listke za opombe. Velikokrat sem uporabil isti listek do te mere, da je bil zapolnjen vsak prazen prostor. Posledično sem se pri prepisu podatkov v evidenčni list soočil z oteženim interpretiranjem prej omenjenih podatkov.

Naročanje storitve je vedno potekalo tako, da je kmet oddal naročilo pri traktoristu z balirko, ta pa je nato meni sporočil seznam in vrstni red strank ter lokacije baliranja. Tako včasih nisem vedel, kdaj bova končala z baliranjem za tisti dan.

1.3 Pričakovani rezultati

Aplikacija je namenjena za uporabo traktoristu, ki dela bale, kot tudi tistemu, ki jih ovija. Sama aplikacija omogoča vnos osnovnih podatkov o stranki oziroma omogoča urejanje obstoječih podatkov. Prav tako omogoča uporabniku ustvarjati seznam strank, ki želijo na določen dan imeti po-balirano travo, slamo ali kaj drugega. Glavna aktivnost aplikacije prikazuje spored strank za trenutni dan in omogoča naknadno dodajanje ali brisanje strank. Po opravljenem baliranju omogoča traktoristu vnos podatkov o baliranju. V primeru, da kmetovalec ne poravna obveznosti

koriščenja storitve, se kmetovalca prikaže v posebnem seznamu z ostalimi dolžniki. Iz seznama dolžnikov se sme stranke odstraniti le ob naknadni poravnavi vseh dolgov. Aplikacija omogoča tudi ustvarjanje evidence (ob različnih kriterijih) v xml formatu, v primeru, če se aplikacija uporablja v podjetju, ki izdaja potrdila o plačilu storitve.

2 PAMETNI TELEFON

Leta 1992 je podjetje IBM poimenovalo prvi pametni telefon z imenom Simon (glej Slika 1), ki je bil leto kasneje kupcem na voljo preko podjetja BellSouth. Poleg navadnih funkcij mobilnega telefona je imel tudi koledar, imenik, svetovno uro, računalo, beležko, elektronsko pošto, igre in možnost prejemanja oziroma pošiljanja sporočil preko faksa. Simon ni imel tipkovnice, ampak so uporabniki za izbiranje številk uporabili zaslon na dotik in preko njega s prsti ali posebnim pisalom dodajali oziroma urejali tekst. V današnjem času bi bil Simon tehnološko zelo zaostali telefon, vendar pa je leta 1993 veljal za zelo naprednega.

Slika 1: Prvi pametni telefon Simon

Pametni telefon je mobilni telefon, ki v primerjavi s sodobnimi osnovno-funkcijskimi telefoni ponuja naprednejše računalniške sposobnosti in povezljivost. Za osnovno-funkcijske telefone je značilno, da so zmožni poganjati aplikacije, ki so razvite na Java ME platformi. Medtem pa pametni telefoni dopuščajo uporabniku, da sam naloži in zaganja zahtevnejše aplikacije. Te naprave lahko smatramo kot osebne žepne računalnike z dodanimi funkcijami mobilnega telefona, saj so ti telefoni navadni računalniki, vendar veliko manjši. Slika 2 prikazuje nekaj primerov današnjih pametnih telefonov.

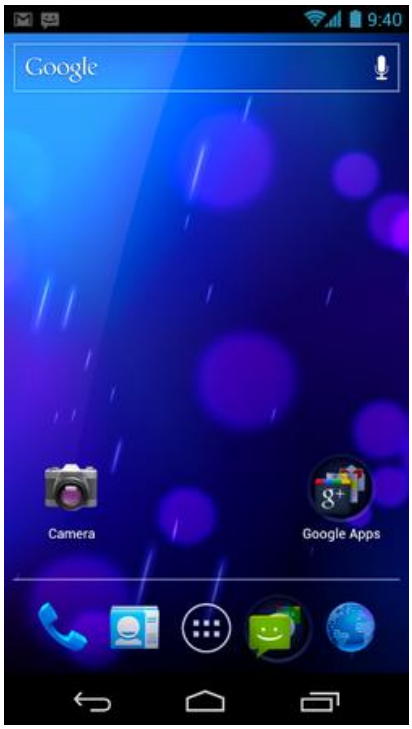
Slika 2: Primeri današnjih pametnih telefonov

Zaradi vse večje priljubljenosti pametnih telefonov, postajajo le-ti predmet napadov z zlonamerno programsko opremo (malware). Pogosto se ta distribuira preko aplikacijskih trgovin kot so Android Market, Apple App Store, Windows Phone Marketplace idr. V nekaterih primerih je omenjena programska oprema skrita v piratskih različicah zakonite aplikacije, ki se nato širijo preko aplikacijske trgovine tretje osebe. Do tveganja prihaja tudi preko tako imenovanih »napadov s posodobitvijo«, kjer se legitimna aplikacija kasneje spremeni, tako da se ji doda komponenta s škodljivo kodo. Tipična zlonamerna oprema za pametne telefone izkorišča ranljivosti platforme, ki omogoča, da nepooblaščene osebe pridobijo korenit dostop do naprave v ozadju. Z uporabo tega dostopa, potem zlonamerna oprema namesti dodatne programe, ki ciljajo na komunikacije, lokacije ali pridobivanje drugih osebnih podatkov. Običajna oblika škodljive programske opreme na mobilnih telefonih je trojanski SMS, ki nevede teče v ozadju zakonite aplikacije in pošilja premium sporočila SMS. Ta sporočila povzročijo tako visoke stroške telefonskega računa, da jih lastnik ne more povrniti [3].

Najboljši način za zmanjšanje ranljivosti proti malware napadom je redno nameščanje najnovejše različice operacijskih sistemov, ki vključujejo varnostne popravke. Prav tako lahko napravo zaščitimo tudi z namestitvijo antivirusnih aplikacij, katere so se začele pojavljati v aplikacijskih trgovinah.

2.1 Operacijski sistem Android

Operacijski sistem Android (glej Slika 3) je namenjen mobilnim napravam kot so pametni telefoni in tablični računalniki. Temelji na Linux jedru, bogatem uporabniškem vmesniku, aplikacijami za končnega uporabnika, vsebuje pa tudi vse funkcionalnosti navadnega mobilnega telefona, multimedijsko podporo in še veliko več. Številne komponente operacijskega sistem so napisane s programskim jezikom C oziroma C++, medtem ko so uporabniške aplikacije razvite v Javi. Operacijski sistem Android je prvotno razvilo podjetje Android Inc in ga je leta 2005 prevzel Google Inc. Sedaj ga razvija organizacija Open Handset Alliance, ki jo vodi Google [4]. OHA je zavezništvo skoraj 84 organizacij (podjetja, ki se ukvarjajo z razvojem strojne in programske opreme ter telekomunikacij), ki so zavezani za »boljši« in bolj »odprt« mobilni telefon na trgu.



Slika 3: Operacijski sistem Android - Ice Cream Sandwich

Pomembni ponudniki operacijskih sistemov na trgu pametnih telefonov so še: Symbian (Nokia), BlackBerry OS (RIM), iPhone (Apple), Windows Phone 7 (Microsoft) [3].

V nadaljevanju je opisan hiter potek pomembnih komponent operacijskega sistema Android [5]:

- Jedro Linuxa zagotavlja temeljno plast abstrakcije strojne opreme, kot tudi osnovne storitve kot so upravljanje s procesi, spominom in datotečnim sistemom. Tu so tudi implementirani gonilniki za strojno opremo kot sta Wi-Fi in Bluetooth. Sklad Androida je fleksibilno

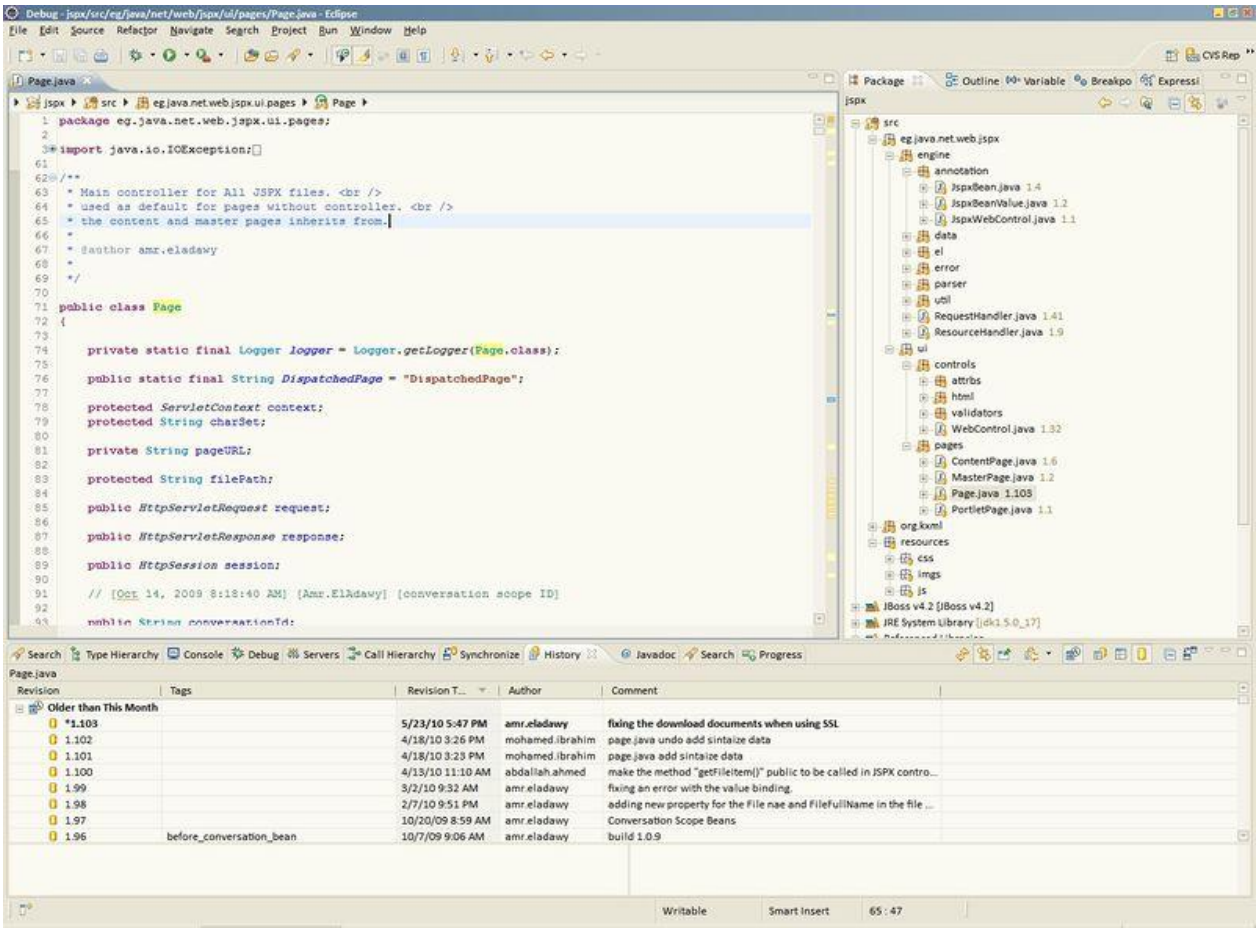
načrtovan z veliko izbiro dodatnih komponent, ki se v veliki meri zanašajo na razpoložljivost strojne opreme določene naprave. Te komponente vključujejo funkcije kot so zaslon na dotik, kamere, GPS sprejemniki in merilci pospeška.

- Pomembne knjižnice kot so WebKit za poganjanje internetnih brskalnikov, SQLite za enostavno uporabo podatkovnih baz, SGL in OpenGL ES za grafično podporo 2D, 3D in animacij, PacketVideo's OpenCORE in Googlovo ogrodje Stagefright za avdio in video podporo in SSL.
- paleto menedžerjev, ki zagotavljajo storitve: aktivnosti in pogledov, okna (Windows), telefonije, virov in lokacijskih storitev.
- v času izvajanja okolja zagotavlja glavne pakete Jave za skoraj popolno funkcionalnost okolja Jave, in Dalvik VM, ki uporablja storitve Linux jedra za zagotavljanje okolja v katerem gostujejo aplikacije Android.

3 UPORABLJENA ORODJA IN TEHNOLOGIJE PRI RAZVOJU

3.1 Eclipse

Eclipse je večjezikovno razvojno okolje za razvoj programske opreme, ki vključuje integrirano razvojno okolje (IDE) in razširljiv sistem vtičnikov (plug-ins). Slika 4 tako prikazuje razvojno okolje Eclipse IDE.



Slika 4: Razvojno okolje Eclipse IDE

V večji meri je napisan s programskim jezikom Java in se uporablja za razvoj aplikacij za Javo, z nameščenimi vtičniki pa tudi za druge programske jezike kot so Ada, C, C++, Python, Perl idr. Izvajalno okolje Eclipse temelji na Equinox, ki je implementacija temeljnih specifikacij ogrodja OSGi (Open Services Gateway initiative framework). Z izjemo majhnega jedra, je vse v Eclipsu vtičnik. To pomeni, da je vsak vtičnik enako integriran z Eclipsom kot vsi ostali in v tej zvezi so vsi elementi »ustvarjeni enako«.

Eclipse SDK vključuje Eclipse Java Development Tools (JDT), ki ponuja IDE z vgrajenim prevajalnikom Java in polni model izvornih datotek Java. To omogoča napredne tehnike tako imenovanega refactoringa in analiziranje kode. IDE prav tako uporablja »workspace«, ki v tem primeru predstavlja niz metapodatkov nad datotečnim sistemom, in omogoča zunanje spremembe datotek dokler se ustrezni vir (resource) kasneje ne osveži v delovnem prostoru.

Razvojno okolje Eclipse je bilo razvito iz dolge linije IBMovih izdelkov razvojnih okolij - IBM VisualAge za Smalltalk ™ in IBM VisualAge za Java. IBM VisualAge Micro Edition ™ je bil prvi resen izdelek in nasploh zelo uspešen. Vendar pa se je za tretje osebe izkazalo, da je težko razširiti izdelek z novimi deli predvsem iz dveh razlogov. Prvi razlog je ta, da načrtovalci pri načrtovanju niso imeli v mislih komponentnega modela, drugi razlog pa je ta, da je bil izdelek zaprto-kodni. Tako se je majhna skupina strokovnjakov odločila, da združijo izkušnje iz prejšnjih let v oblikovanju in izvajanju razvojnih okolij, rezultat pa je bil Eclipse, platforma zasnovana iz nič, kot platforma za povezovanje razvojnih orodij. Ta je partnerjem omogočila enostavno razširitev izdelkov narejenih na njej, z uporabo vtičnih mehanizmov, ki jih zagotavlja platforma [6].

3.2 Android SDK

Aplikacije za platformo Android se razvijajo s programskim jezikom Java in z uporabo Android Software Development Kit (SDK), obstajajo pa tudi druga orodja. Android SDK vsebuje obsežen sklop razvojnih orodij med katere vključujemo razhroščevalnik, knjižnice, emulator telefona (glej Slika 5), dokumentacijo, vzorčno kodo in razne vaje. Trenutno podprte platforme za razvoj vključujejo računalnike z operacijskim sistemom Linux (vseh modernih desk-top distribucij), Mac OS X 10.4.9 ali novejši ter Windows XP ali novejši. Uradno podpira integrirano razvojno okolje (IDE) Eclipse (trenutno 3,5 ali 3,6) z vtičniki Android Development Tools (ADT). Razvijalci lahko kljub temu uporabljajo vsak urejevalnik besedila za urejanje Java in XML datotek. Nato pa z uporabo orodja ukazni poziv (na računalniku morata biti nameščena Java Development Kit in Apache Ant) ustvarijo, zgradijo in odpravljajo napake v aplikaciji Android. SDK podpira tudi starejše različice platforme Android v primeru, če želijo razvijalci razvijati aplikacije starejše naprave [4].

Slika 5: Simulator operacijskega sistema Android

3.3 Java

Java je objektno-usmerjen programski jezik, ki ga je razvil James Gosling s sodelavci iz Sun Microsystems (v letu 2010 je prišlo do združitve z Oracle Corporation). Velik del sintakse izhaja iz C in C++, vendar ima enostavnejši objektni model in manj nizko nivojskih funkcij. Trenutno je eden izmed najbolj priljubljenih programskih jezikov v uporabi, zlasti za razvoj spletnih aplikacij, odjemalec-strežnik in mobilnih aplikacij.

Ena izmed značilnosti Jave je prenosljivost, ki pomeni, da mora računalniški program, napisan v jeziku Java, delovati podobno kot na kateremkoli ekvivalentno zmogljivem računalniku. To se doseže z zbiranjem kode jezika Java v vmesno predstavitev imenovano Java bytecode, namesto da se neposredno prevede v strojno kodo za določeno platformo. Ukazi Java bytecode so podobni strojnim ukazom, vendar so namenjeni interpretaciji navideznega stroja (Virtual Machine). Končni uporabniki pogosto uporabljajo Java Runtime Environment (JRE), ki je nameščen na

njihovem računalniku za samostojne aplikacije Java, ali v spletnem brskalniku za aplikacije Java applet.

Programi pisani v Javi so počasnejši in potrebujejo več spomina kakor tisti, ki so pisani na primer v C-ju. Z uvedbo zbiralnika Just-in-Time (JIT), v različici 1.1, se je hitrost izvajanja programov Java bistveno izboljšala. Poleg tega je omenjena različica vsebovala tudi dodatne funkcije, optimiziran pa je bil tudi sam navidezni stroj [7].

3.4 SQLite

SQLite je vgrajen sistem za upravljanje z relacijskimi podatkovnimi bazami, vsebovan v majhni knjižnici programskega jezika C. Zagotavlja večji del standarda SQL, z uporabo dinamično in šibko natipkano SQL sintakso, ki ne zagotavlja integritete domene. SQLite lahko z večopravilnostjo opravi vzporedno več operacij branja, pisanje pa se lahko izvajajo samo zaporedno.

Za razliko od sistemov odjemalec-strežnik za upravljanje s podatkovnimi zbirkami, SQLite nima samostojnih procesov s katerimi komunicira uporabniški program. Namesto tega je knjižnica povezana v SQLite in s tem postane sestavni del aplikacije. Program uporablja funkcionalnosti SQLite s preprostimi klici funkcij, ki zmanjšajo latenco pri dostopu do podatkovne zbirke (klici funkcij v enem samem procesu so bolj učinkoviti kot medprocesne komunikacije). SQLite hrani celotno zbirko podatkov (definicije, tabele, indekse in podatke), v eni cross-platform datoteki na gostiteljevi napravi [8].

4 APLIKACIJA BALIRANJE TRAVE

Razvoj aplikacije je potekal v večih stopnjah. Najprej smo pridobili osnovne podatke na podlagi opazovanja traktorista pri baliranju. Potem smo se sami postavili v njegovo vlogo in razmislili katere podatke bi še potrebovali za čim bolj učinkovito delo. Čeprav smo naučeni, da se aplikacija začne z razvojem podatkovnega modela in s postavitvijo zbirke podatkov, smo na začetku uporabili svoj pristop in začeli z razvojem uporabniškega vmesnika. Razlog za takšen postopek izvira iz težav pri postavitvi zbirke podatkov, katere smo s trdo odločnostjo tudi rešili. Po postavitvi zbirke smo se lahko držali naučene prakse programiranja v podjetju Agito. Tako smo najprej implementirali logiko za manipulacijo s podatkovno bazo, zatem še logiko za obdelavo podatkov in na koncu prikaz podatkov uporabniku. Ob koncu razvoja aplikacije Baliranje trave nam je preostalo le še testiranje omenjene aplikacije na mobilni napravi.

4.1 Podatkovni modul

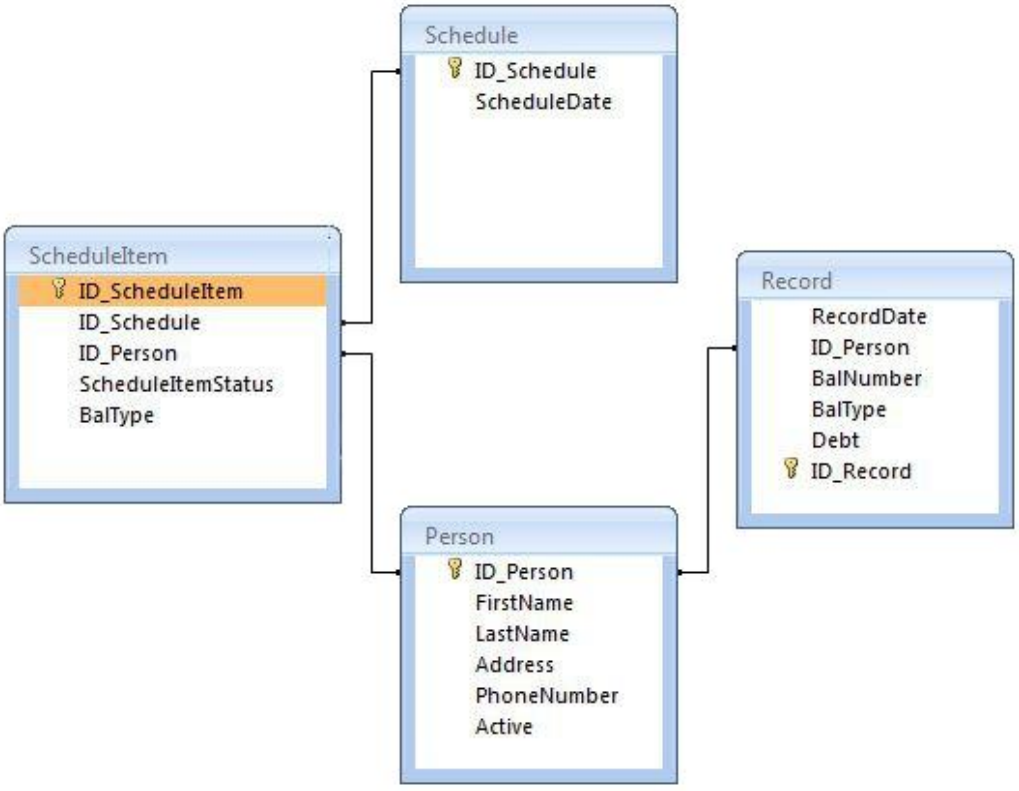
4.1.1 Zbirka podatkov

Za postavitev podatkovne baze nismo imeli na razpolago kakšnega naprednega orodja za kreiranje baze. V primeru, da bi imeli na razpolago kakšen tak program, bi se soočili s težavo uvažanja bazne datoteke v projekt aplikacije oziroma kako jih namestiti na mobilno napravo.

Android ima implementiran razred SQLiteOpenHelper, s katerim smo ustvarili in upravljali s podatkovno bazo SQLite. Omenjeni razred je bilo potrebno samo razširiti in prepisati metodo onCreate z ukazi za ustvarjenje tabel z ustreznimi relacijami med tabelami. Preden lahko preberemo kakšen podatek iz baze, ga je najprej potrebno vanjo zapisati, kar smo dosegli na dva načina. Prvi je z metodo execSQL, ki za parameter potrebuje običajen SQL insert stavek in je dinamično sestavljen (imena stolpcev in vrednosti). Drugi način je bolj preprost, in sicer z uporabo metode insert, ki potrebuje ime tabele in podatke, ki jih želim zapisati. Poleg teh dveh metod za manipulacijo s podatki poznamo še metodi update in delete. Metoda update se uporablja za posodabljanje obstoječih podatkov in sprejme enake parametre kot metoda insert. Zadnja metoda za manipulacijo s podatki v zbirki je metoda delete, ki ima prvi parameter ime tabele, z drugim parametrom pa pove kaj naj izbriše iz tabele.

Sedaj je že nekaj podatkov v bazi, ki jih lahko preberemo in tudi to lahko naredimo na dva načina. Prvi način se uporablja za branje podatkov iz ene tabele z metodo query, drugi pa za branje iz več tabel z združevanjem tabel (SQL ukaz JOIN). V obeh primerih dobimo podatke v

vmesniku Cursor in s pomočjo ekstraktorjev iz njega pridobimo podatke ter jih nastavimo entitetam.

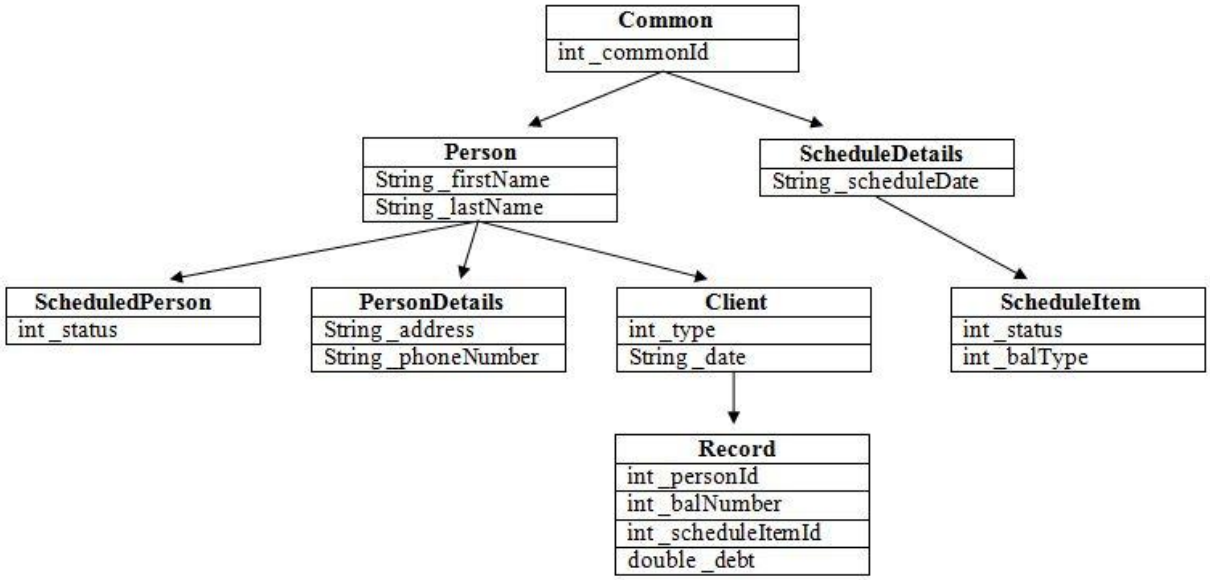


Slika 6: Model zbirke podatkov

4.1.1 Entitete

Entitete si lahko predstavljamo kot objekte, ki vsebujejo polja ali lastnosti za vsak stolpec naveden v poizvedbi (glej Slika 7). Vsaka entiteta mora imeti unikaten identifikator objekta ali primarni ključ, ki je lahko enostaven ali sestavljen in omogoča iskanje določene instance entitete. Bistvena razlika med shranjevanjem podatkov v polja oziroma lastnostmi je v tem, da se pri poljih med izvajanjem programa neposredno dostopa do vrednosti spremenljivk oziroma nastavljajo vrednosti spremenljivk, pri lastnostih pa preko metod setter in getter. Torej ena instanca entitete predstavlja eno vrstico v poizvedbi in iz tega sledi, da celotno poizvedbo predstavlja podatkovna struktura Vektor<entiteta>. Razloga za njegovo uporabo sta dva in prvi izhaja iz samega delovanja Vektorja, saj ta namreč omogoča branje elementov v takšnem zaporedju kot so bili dodajani. Drugi razlog se nanaša na poizvedbe, katere omogočajo sortiranje po različnih stolpcih in s tem prihranimo nekaj procesorskega časa.

Njihova uporaba pride do izraza pri ustvarjanju dinamičnih seznamov ali prirejanju komponent, ki zahtevajo dodatno manipulacijo podatkov.



Slika 7: Hierarhija entitet

4.1.2 Ekstraktorji

Ekstraktorji so pomožni razredi, s katerimi iz rezultata poizvedbe (Cursor) izluščimo ustrezne podatke in jih nastavimo ustreznim entitetam. Njihova hierarhija je identična hierarhiji entitet.

Postopek izluščenja podatkov se opravi z dvema metodama. Najprej se v metodi getColumnId na podlagi imen stolpcev definiranih v poizvedbi, nastavijo identifikatorji stolpcev poizvedbe. Nato se v metodi getData najprej ustvari nova instanca entitete. S pomočjo metod za branje določenega primarnega podatkovnega tipa izluščimo vrednosti posameznega stolpca ter preko setter metod nastavimo lastnosti entitete. Na koncu metoda vrne instanco novo ustvarjene entitete.

4.2 Poslovna logika

V tem podpoglavju so opisane komponente, ki tečejo v ozadju (AsyncTask in Service) ali obdelujejo podatke (BroadcastReceiver), uporabniku pa se prikažejo le kot obvestilo v vrstici z obvestili (Notifications).

4.2.1 Asinhrona naloga

Razred AsyncTask omogoča pravilno in enostavno uporabo niti uporabniškega vmesnika. Ta omogoča izvajanje operacij v ozadju in objavljanje rezultatov na uporabniški vmesnik, brez kakršne koli manipulacije niti ali handlerjev. Asinhrona naloga je definirana s tremi generičnimi tipi (Params, Progrss in Result) in v štirih korakih (onPreExecute, doInBackground,

onProgressUpdate in onPostExecute). Da se ta razred lahko uporabi, ga je potrebno razširiti in prepisati vsaj metodo doInBackground ter v večini primerov tudi metodo onPostExecute.

V aplikaciji se ta razred uporablja na nivoju poslovne logike. Uporablja se tako, da se razširi kot abstrakten zasebni razred (MyTask), v katerem se prepišeta samo metodi onPreExecute (za začetek prikazovanja ProgressBar dialoga) in onPostExecute (za zaključitev prikazovanja ProgressBar dialoga). Nato smo v vsaki metodi poslovne logike implementirali BackgroundTask, kot anonimni notranji razred in v doInBackground opravili klic ustrezne metode na podatkovnem nivoju. S tem pristopom se nabere kar nekaj vrstic programske kode, obstaja pa način programiranja, tako imenovani reflection, s katerim se lahko zmanjša število vrstic. Vendar ima reflection nekaj slabosti (izvajanje operacij ni ravno hitro in nima optimizacije kode), ki pridejo do poudarka pri aplikacijah, kjer je pomembna hitrost [9].

4.2.2 BroadcastReceiver

Sprejemnik oddajanja je komponenta, ki se odziva na celotni ravni sistema na določeno obvestilo. Ta komponenta ne prikaže ničesar na uporabniški vmesnik, ampak prikaže le obvestilo v vrstici stanja ob določenem dogodku in je namenjena za opravljanje minimalne količine dela. Na primer, lahko začne storitev za opravljanje časovno bolj potratne naloge [9].

V tej aplikaciji smo to komponento uporabili za samodejno dodajanje strank v spored s pomočjo sporočil SMS. Ob prejetju sporočila in pod pogojem, da je sporočilo v vnaprej določenem formatu s ključno besedo, se začne izvajati logika sprejemnika. V primeru, da pride do napake pri izvajanju, se v vrstici stanja prikaže obvestilo in uporabnik ročno doda stranko v spored, v nasprotnem primeru se prikaže obvestilo o uspešno opravljenem dodajanju.

4.2.3 Service

Storitev (Service) je ena izmed glavnih komponent aplikacije Android, ki lahko opravlja dolgotrajne operacije v ozadju in ne zagotavlja uporabniškega vmesnika. V večini primerov storitev zaženeta aktivnost (Activity) ali sprejemnik (BroadcastReceiver) in se izvaja naprej, tudi če uporabnik aplikacijo zapre [9].

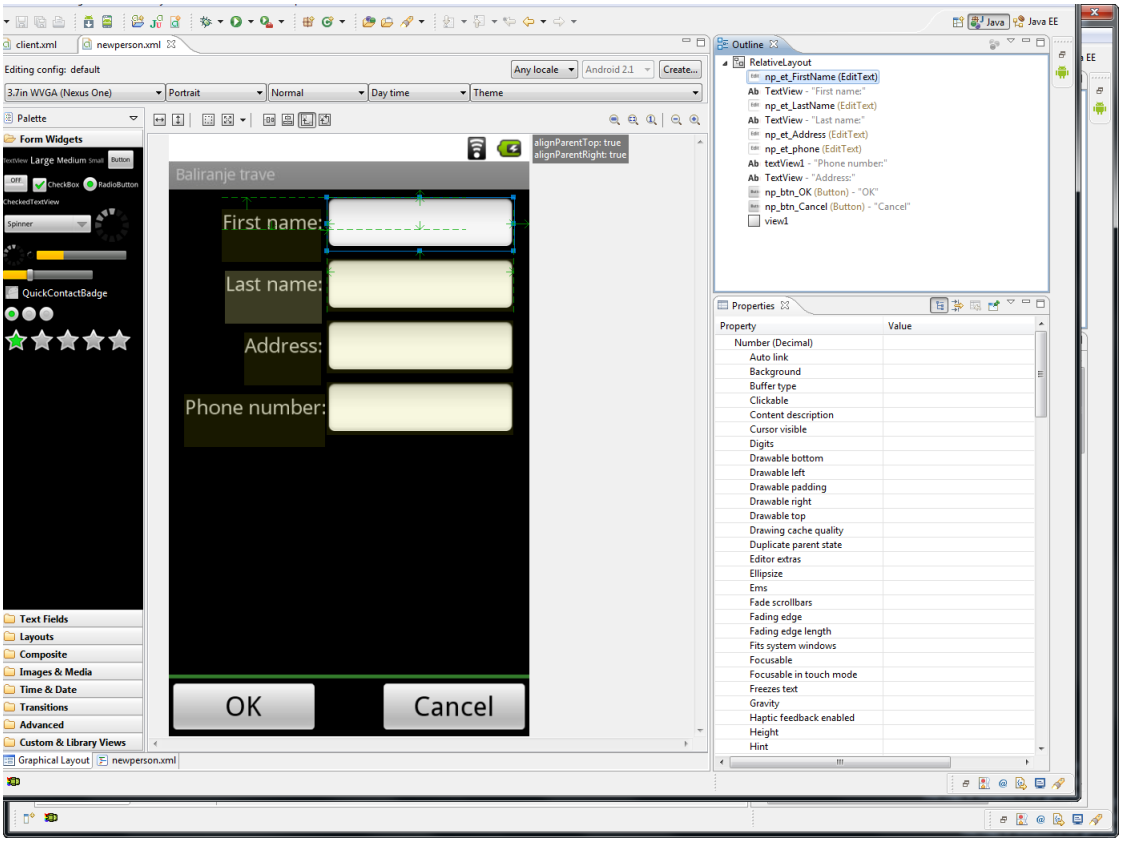
Ta komponenta se v aplikaciji uporablja za brisanje nepotrebnih podatkov o sporedu, s čimer prihranimo malo internega spomina, sproži pa se v zgodnjih jutranjih urah. Razlog za tako pozno proženje je enostaven, saj se lahko delo traktorista v času baliranja trave zavleče tudi do tretje ure zjutraj. V takšnem primeru mora imeti traktorist na voljo še spored prejšnjega dneva.

4.3 Uporabniški vmesnik

Iz izkušenj pri razvoju aplikacij Windows v podjetju Agito d.o.o., se je tudi za to aplikacijo izkazalo, da razvoj uporabniškega vmesnika zavzame največ časa v primerjavi z implementacijo ostale logike. V nadaljevanju sta opisana dva načina izdelave uporabniškega vmesnika, katera smo tudi uporabili pri razvoju.

Prvi način izdelave je ustvarjenje vseh gradnikov s programsko kodo. Ta način je zelo počasen in časovno zahteven za programerja, vendar je zelo uporaben pri dinamičnem kreiranju komponent za prikaz na uporabniškem vmesniku.

Drugi način, ki je veliko hitrejši in prijaznejši do programerja, je tako imenovani povleci in spusti (drag and drop). S tem načinom programer iz seznama komponent povleče komponento in jo spusti na zaslon, ter z njo poljubno manipulira z nastavljanjem lastnosti. Vse komponente in njihove lastnosti, ki jih je programer nastavil se zapišejo v datoteko XML. Da se prikaže kar je programer naredil z grafičnem urejevalnikom, mora v metodi onCreate, ki se vedno prepiše pri razširjanju razreda Activity, nastaviti katera razporeditev (layout) naj se prikaze ob zagonu aktivnosti.



Slika 8: Grafični urejevalnik uporabniškega vmesnika

4.3.1 Aktivnost »Spored«

Ob zagonu aplikacije se prikaže aktivnost »Spored«, katera vsebuje samo tri elemente: naslov aktivnosti, mrežni pogled in v njem »FarmerView«. Naslov je sestavljen iz besede »Spored:« in trenutnega datuma. Elementi v mreži so razporejeni po principu vrste FCFS (prvi, ki pokliče ima najprej po-balirano) in v treh stolpcih. »FarmerView« je prirejena komponenta, kateri se na podlagi prebranih podatkov iz zbirke podatkov nastavijo atributi, ki določajo njen prikaz. Ti podatki so ime, priimek in stanje, ki služi kot indikator o že opravljenem baliranju pri določenem kmetu.

S pritiskom na element se prikaže aktivnost »Stranka«, katera vsebuje še meni in se prikaže, ko pritisnemo na tipko menu. V meniju obstajajo tri možne preusmeritve (glej Slika 9). Prva preusmeritev je preusmeritev na aktivnost »Ustvari spored«, druga na aktivnost Izvoz podatkov in nastavitve, kot zadnja pa je mogoča preusmeritev na aktivnost »Seznam dolznikov«.

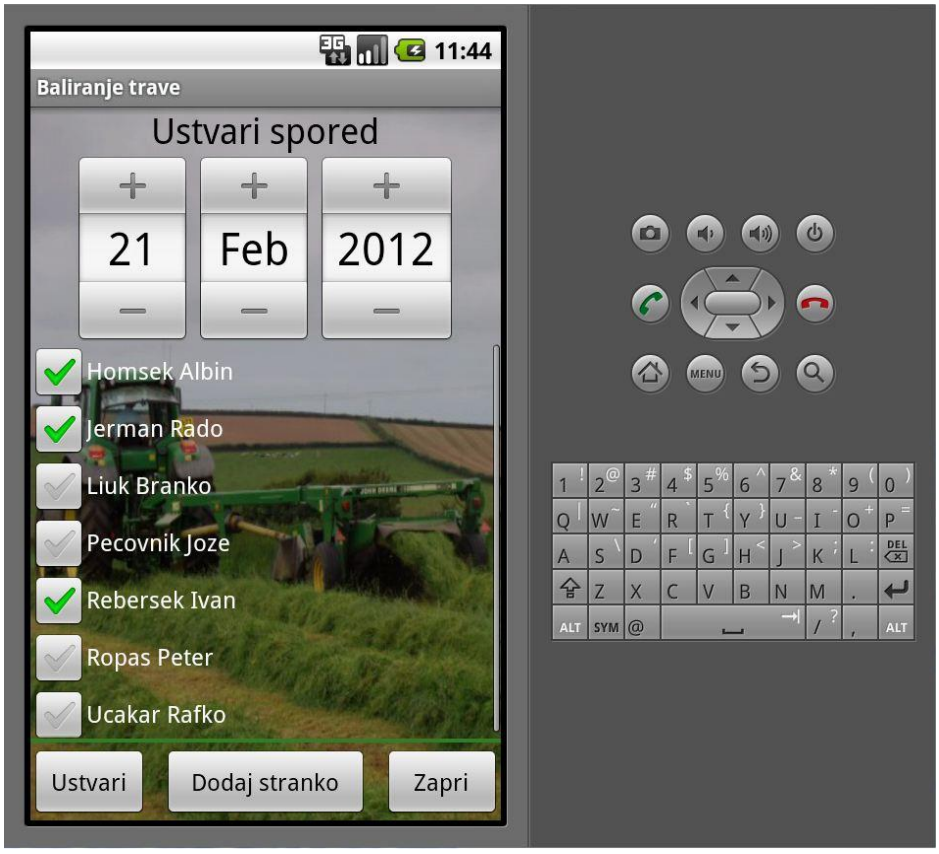
Slika 9: Aktivnost »Spored«

4.3.2 Aktivnost »Ustvari spored«

Aktivnost »Ustvari spored« je namenjena dodajanju strank v obstoječ spored oziroma ustvarjanju novega. Vendar pa to ni edini način za dodajanje oziroma ustvarjanje sporeda. Aplikacija vsebuje funkcionalnost, ki na podlagi sporočila SMS s ključno besedo samodejno ustvari spored in/ali doda stranko v spored.

Aktivnost »Ustvari spored« je sestavljena iz treh delov poleg naslova aktivnosti, kar prikazuje Slika 10. V zgornjem delu je DatePicker, ki služi za izbiro datuma sporeda. Srednji del je namenjen seznamu potrditvenih polj, ki se ustvari na podlagi prebranih podatkov shranjenih v zbirki podatkov. Spodnji del pa vsebuje tri gumbe. Prvi je gumb »Ustvari«, ki v zbirko zapiše vse izbrane kmete, torej tiste, ki želijo na izbran datum imeti po-balirano travo oziroma slamo. V primeru, da kmeta ni v seznamu, se s pritiskom na gumb »Dodaj stranko« sproži postopek ustvarjanja nove osebe. Zadnji gumb je gumb »Preklici«, ki zaključi trenutno aktivnost in se vrne na »Spored«.

Dolgi klik na potrditveno polje preusmeri uporabnika na aktivnost »Dodaj novo stranko«, ki je v tem primeru namenjena za urejanje obstoječih podatkov izbrane stranke.

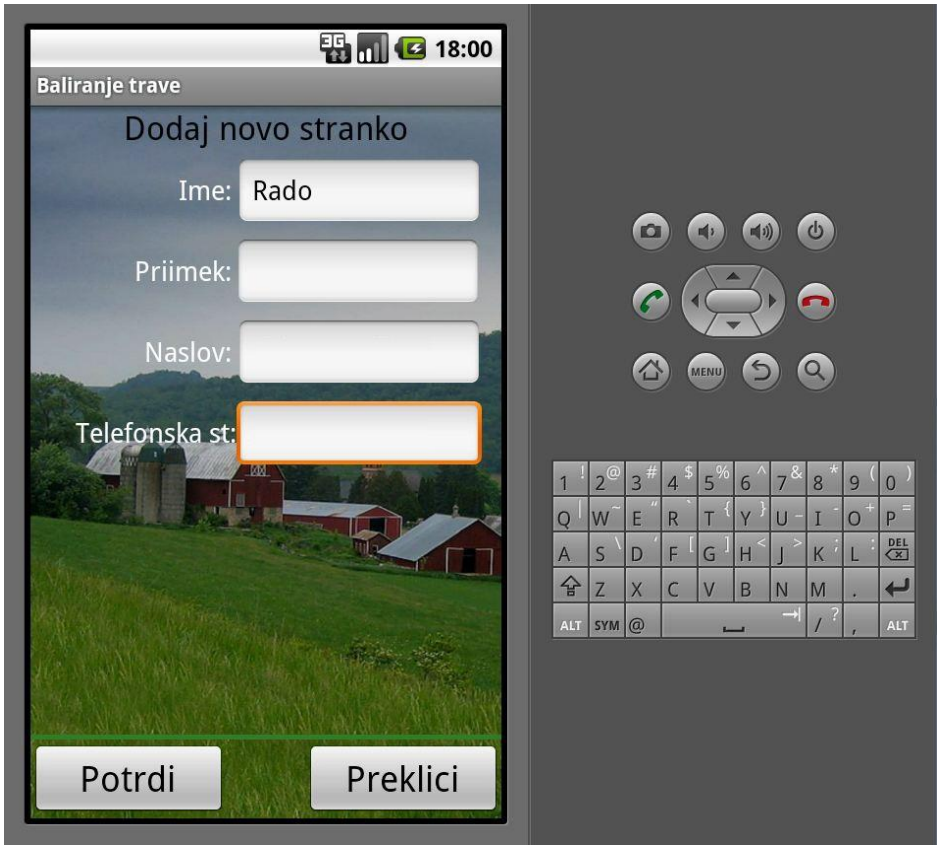


Slika 10: Aktivnost »Ustvari spored«

4.3.3 Aktivnost »Dodaj novo stranko«

Aktivnost »Dodaj novo stranko« (Slika 11) je namenjen vnosu podatkov o stranki v aplikacijo. Vsebuje štiri vnosna polja (EditText), ki morajo biti obvezno izpolnjena (ime, priimek, naslov in telefonska številka). V primeru, da uporabnik pozabi oziroma vnese napačno obliko podatka (telefonska številka), se prikaže obvestilo o manjkajočem oziroma neveljavnem podatku, kar predstavlja prvi primer uporabe te aktivnosti. Drugi primer uporabe aktivnosti pa predstavlja urejanje podatkov obstoječe stranke, kar uporabnik doseže z dolgim klikom nekega elementa v seznamu strank na modulu »Ustvari spored«.

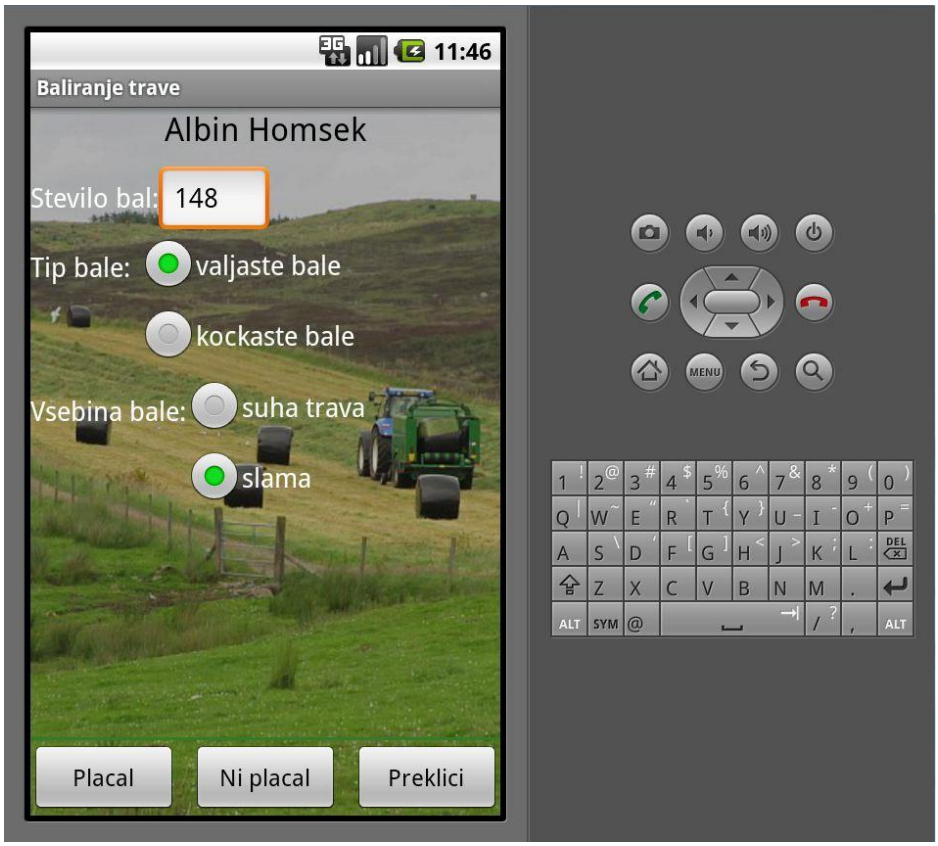
S pritiskom na gumb »Potrdi« se sproži preverjanje podatkov in zapis v zbirko podatkov. Potem vrne uporabnika na »Ustvari spored«, kjer se sproži osveževanje seznama strank. Gumb »Preklici« vrne uporabnika na »Ustvari spored«.



Slika 11: Aktivnost »Dodaj novo stranko«

4.3.4 Aktivnost »Stranka«

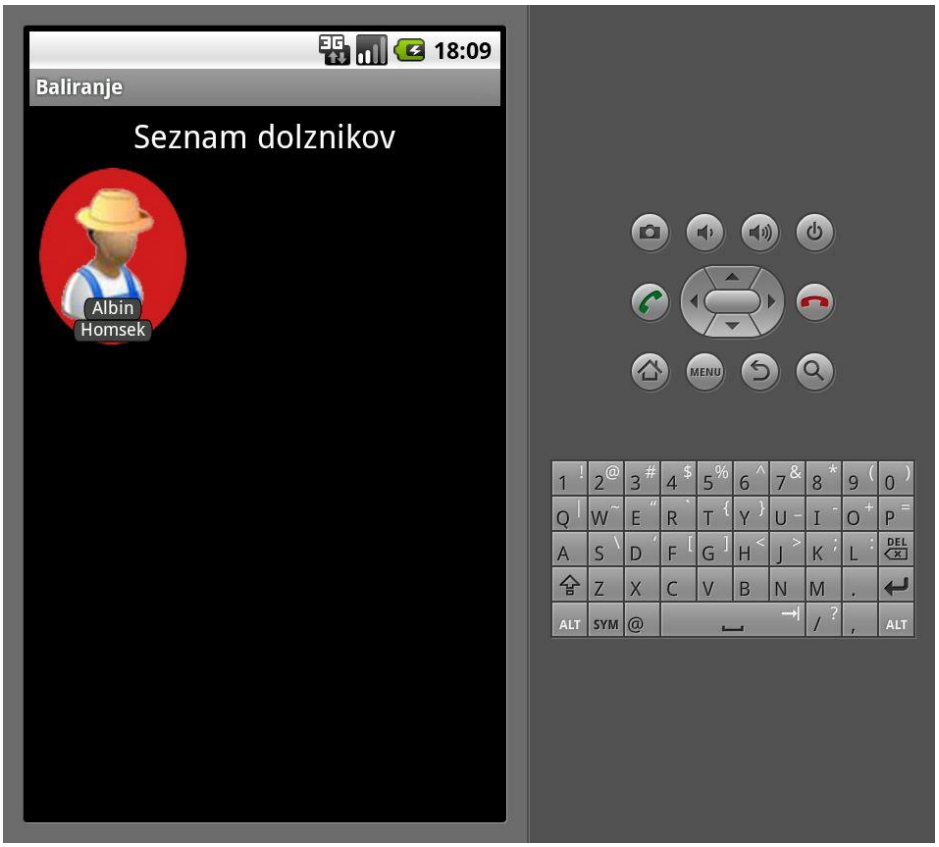
Traktorist uporabi aktivnost »Stranka« po opravljenem baliranju pri kmetu. Ta prikazuje ime in priimek stranke, tako da traktorist ne vnaša podatkov na slepo za določenega kmeta. Aktivnost (glej Slika 12) ima na voljo tudi vnosno polje v katerega uporabnik vnese število bal, ki jih je naredil ter določi tip in vsebino bal. Balira se lahko sveže oziroma ovenelo travo, suho travo ali slamo. Podatek o tipu baliranja ni samo informativne narave, ampak določa ceno storitve. Bale z ovenelo travo je potrebno poviti s posebno folijo, da se pripravijo pogoji za potek anaerobnih biokemičnih reakcij, medtem ko pri ostalih tipih bal ni potrebno ovijanje bal v folijo in tako to predstavlja manjši strošek. Na koncu uporabnik v primeru, da je stranka poravnala stroške baliranja pritisne na gumb »Placal «, v nasprotnem primeru pritisna na »Ni placal« in stranka se doda v seznam »Dolzniki «.



Slika 12: Aktivnost »Stranka«

4.3.5 Aktivnost »Seznam dolznikov«

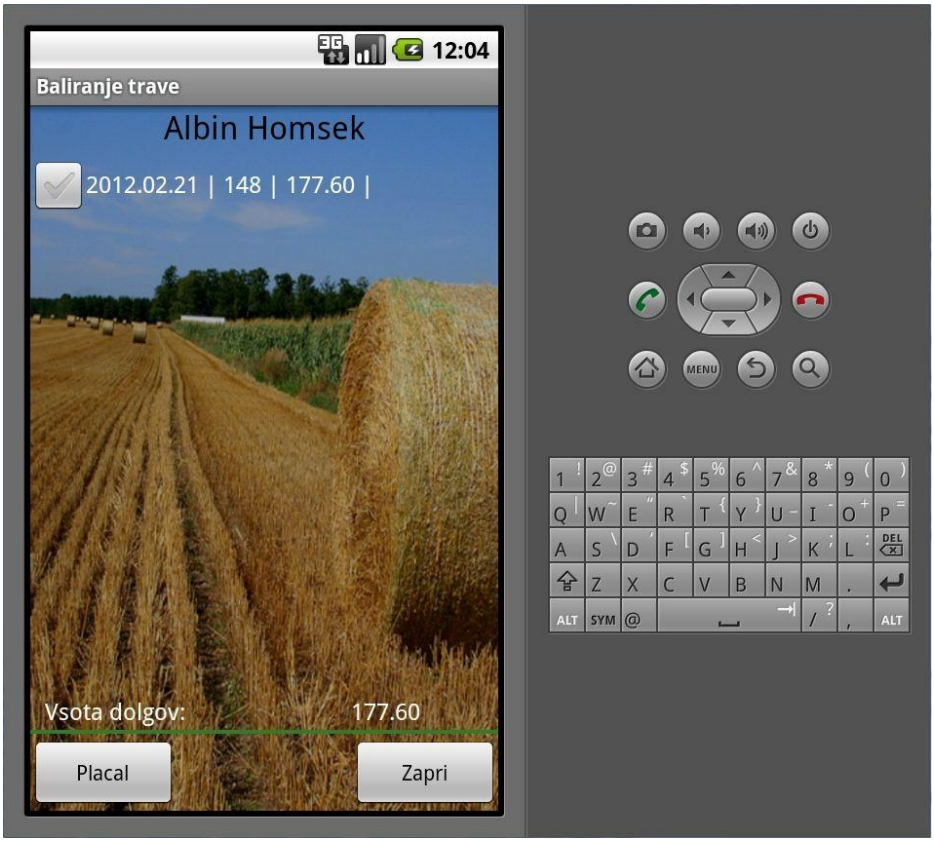
Aktivnost »Seznam dolznikov« (Slika 13) je identičen sporedu, vsaj kar se tiče gradnikov. Razlikuje se v tem, da nima menija in prikazuje stranke, ki imajo neporavnane račune do traktorista oziroma podjetja. S pritiskom na stranko aplikacija preusmeri uporabnika na aktivnost »Podrobnosti dolznika«.



Slika 13: Aktivnost »Seznam dolznikov«

4.3.6 Aktivnost »Podrobnosti dolznika«

Aktivnost »Podrobnosti dolznika« prikazuje seznam dolgov (datum, število bal in dolg), katere stranka še ni poravnala. Na tej aktivnosti se nahajata še dva informativna podatka, ime in priimek ter skupni dolg (glej Slika 14).

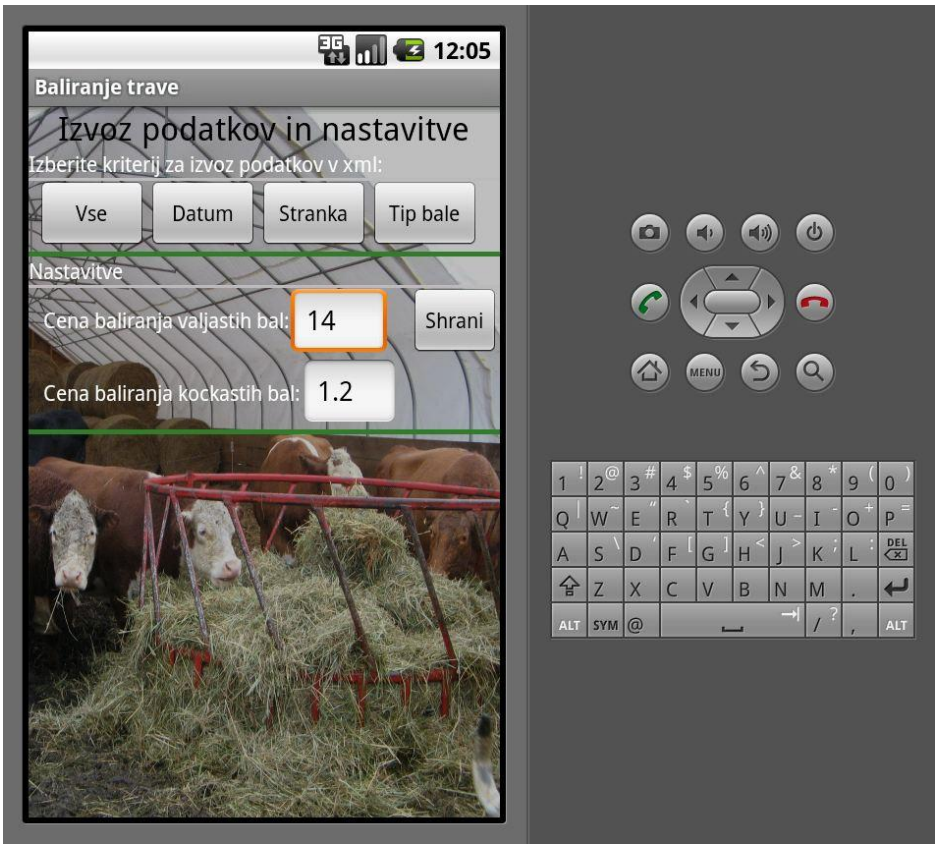


Slika 14: Aktivnost »Podrobnosti dolžnika«

4.3.7 Aktivnost »Izvoz podatkov in nastavitve«

Ta aktivnost prikazuje Slika 15 in se primarno uporablja za izvažanje podatkov o opravljenem baliranju. Z ustreznim pritiskom na enega izmed gumbov, ki predstavljajo kriterije pod katerimi se bo ustvarila xml datoteka s podatki. Prvi kriterij je »Vse«, ki zapiše vse podatke o opravljenem baliranju. Naslednji kriterij je »Datum«, kjer se prikaže pogovorno okno z DatePickerjem ter na podlagi izbranega datuma se izvozijo podatki. V kriteriju »Stranka« se prikaže pogovorno okno s seznamom strank, kjer se na podlagi izbranega elementa s seznama izvede izvoz. Zadnji kriterij je »Tip bale«, ki prikaže pogovorno okno s tipoma bal in z izbiro enega tipa ustvari datoteko xml.

Sekundarna funkcija te aktivnosti je nastavljanje in shranjevanje cen storitev baliranja. Tu uporabnik vnese želene zneske v vnosna polja za posamezni tip baliranja. S pritiskom na gumb »Shrani« se vneseni vrednosti shranita v interni spomin mobilne naprave. Ti vrednosti sta potrebni za računanje zneska storitve na aktivnosti »Stranka«.



Slika 15: Aktivnost »Izvoz podatkov in nastavitve«

4.4 AndroidManifest.xml

Vsaka Android aplikacija mora imeti datoteko AmdrodManifest.xml v svojem korenskem imeniku projekta. V njej so bistvene informacije o aplikaciji, ki jih operacijski sistem Android potrebuje še preden se začne izvajati programska koda (Slika 16).

Pri razvoju aplikacije smo uporabili deset različnih elementov, pri čemer prvi nivo vsebuje tri elemente: pravice uporabnika, aplikacijo in uporabnikov SDK. Pravice uporabnika (user-permission) z nastavitvijo atributov omogočajo aplikaciji Baliranje trave sprejemanje sporočil SMS in pisanje na kartico SD. Element aplikacija (aplication) vsebuje tri od štirih osnovnih gradnikov aplikacije Android (aktivnost, storitev in sprejemnik) in atributa tega elementa nastavljata ime in ikono aplikacije. Zadnji element prvega nivoja je uporabnikov SDK (users-sdk), ki določa minimalno stopnjo Android API. Element storitev omogoča aplikaciji izvajanje opravil v ozadju. Sprejemnik omogoča izvajanje BrodcastReceiverja ob določenem dogodku (npr. prejeto sporočilo SMS). Vse kar želimo prikazati uporabniku gre preko aktivnosti, zato je teh elementov tudi največ v tej datoteki. Aktivnost, ki se prikaže ob zagonu aplikacije, se obvezno nastavi kategorija (»android.intent.category.LAUNCHER«) in akcija (»android.intent.action.MAIN«).

PK-Baliranje Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="js.pk.baliranje"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name="NewPerson">
        </activity>
        <activity android:name="CreateSchedule">
        </activity>
        <activity android:name="ClientUI">
        </activity>
        <activity android:name=".Schedule" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="Settings">
        </activity>
        <activity android:name="DebtList">
        </activity>
        <activity android:name="Debtor">
        </activity>
        <receiver android:name="SMSReceiver" android:enabled="true">
            <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED"></action>
            </intent-filter>
        </receiver>
        <receiver android:name="receiver.AlarmReciever"></receiver>
        <service android:name="service.EraseScheduleItemsService"></service>
    </application>
    <uses-sdk android:minSdkVersion="7" />
</manifest>
```

Slika 16: Datoteka AndroidManifest.xml

4.5 Testiranje

Testiranje je potekalo v dveh delih. Najprej smo testirali na simulatorju, ki je vključen v Android SDK plug-in, med samim razvojem aplikacije. Prvi del tako predstavlja okoli 80 % vsega testiranja. Po implementaciji določene logike smo testirali ali deluje pravilno (prebrani, obdelani in prikazani pravi podatki) in v primeru, da se je pojavila napaka, jo tudi odpravili ter ponovili postopek.

Ta način testiranja ima manjšo pomanjkljivost, saj simulator vzame procesno moč računalnika, ki je nekajkrat večja od procesne moči pametnega telefona.

Drugi del testiranja je potekal na pametnem telefonu HTC Desire z operacijskem sistemom Android 2.2 Froyo. Pri tem testiranju je zlasti prišlo do poudarka delovanja več niti hkrati.

4.6 Možne izboljšave

V poglavju Uporabniški vmesnik je predstavljena prva različica aplikacije Baliranje trave. V času pisanja diplomskega dela so se pojavile nove ideje o morebitnih izboljšavah, da bi bila aplikacija uporabnejša. V nadaljevanju je opisanih le nekaj idej, ki bi lahko bile implementirane v naslednji verziji aplikacije Baliranje.

4.6.1 Zemljevid

Proti koncu razvoja se je pojavila ideja, da bi lahko aplikacija prikazovala zemljevid s pobarvanimi zemljišči posameznega kmeta oziroma naj bi imela aplikacija implementirane funkcionalnosti aplikacije Google Maps, kot so satelitski pogled, izris poti med točkama A in B ter trenutna lokacija uporabnika. Poleg naštetih bi bilo potrebno implementirati še nekaj dodatnih funkcij, kot so na primer pobarvana zemljišča (za vsako stranko drugačna barva) na katerih naj bi se kaj baliralo. V ta namen bi morali verjetno razviti posebno aplikacijo, s katero bi stranka na zemljevidu označila meje svojih zemljišč (samo tista, kjer bi se baliralo). Potrebno bi bilo še razviti aplikacijo za izračun oziroma izris najkrajše poti med zemljišči za posamezno stranko v sporedu. Ta dodatek aplikaciji bi najbolj prišel prav tistim traktoristom, ki okolice ne poznajo do potankosti.

4.6.2 Varnostni mehanizem

V prvi različici aplikacije Baliranje se zgodi samodejno dodajanje stranke v spored na podlagi sporočila SMS, ki ima ranljivost. Mehanizem nima implementirane logike za preverjanje, ali je sporočilo res poslala oseba, ki želi koristiti storitve baliranja. V naslednji različici bi pri dodajanju nove stranke dodatno prikazali naključno generirane kode. Ta se po prvi opravljeni

storitvi zaupa stranki, ki jo nato uporablja za naročanje preko sporočil SMS in potrdi željo po koriščenju storitve.

4.6.3 Aplikacija za naročanje storitev baliranja

Trenutno mora stranka za naročanje storitev točno vedeti format in ključno besedo sporočila, kar zna biti včasih zelo zoprna zadeva, še posebej, če manjka kakšen ključen podatek. V ta namen bi razvili še malo aplikacijo, ki bi kmetom omogočala čim lažje naročanje storitev. Aplikacija bi verjetno vsebovala eno aktivnost, v kateri bi bila vnosna (ime, priimek, naslov in potrditvena koda stranke) in izbirna (datum in tip baliranja) polja za vse podatke, ki jih stranka sedaj spiše v sporočilo SMS. Tako bi aplikacija na osnovi vnesenih podatkov sestavila, zakodirala (varovanje osebnih podatkov) in poslala SMS sporočilo traktoristu, ki opravlja storitev.

5 SKLEPNE UGOTOVITVE

Cilj diplomske naloge je bil razviti aplikacijo, s katero bi bilo traktoristu, ki ima v lasti balirko ali opravlja storitev preko kmetijske zadruge, omogočeno čim enostavnejše opravljanje storitve baliranja. Na začetku nisem imel ravno težke odločitve za katero platformo razviti aplikacijo, ker imam v lasti pametni telefon HTC Desire z operacijskim sistemom Android. Kot prvotno .NET razvijalec sem moral opustiti kar nekaj postopkov programiranja, ki sem jih bil vajen v programskem jeziku C#. Tako sem najprej začel z razvojem uporabniškega vmesnika, saj sem se moral čim prej privaditi na novo razvojno okolje. Nato sem se s pomočjo dokumentacije na spletni strani Android Developers hitro seznanil s samo arhitekturo Androida, življenjskim ciklom aktivnosti in pogovornimi okni ter kako se uporabljajo ostale komponente, uporabljene v razvoju. Poleg dokumentacije so na spletu tudi video vodiči, s katerimi sem si pomagal rešiti marsikateri problem.

Proti koncu razvoja aplikacije sem prišel do spoznanja, da sem uporabil le peščico vseh komponent, ki jih vsebuje Android. Razlog izvira predvsem iz tega, da se platforma konstantno nadgrajuje in se bo zaradi vse večje priljubljenosti tudi v prihodnosti.

KAZALO SLIK

VIRI

[1] (2012) Baliranje. Dostopno na: http://www.virc.si/storitve/baliranje

[2] (2012) Baliranje. Dostopno na: http://www.zamet.si/14.0.html

[3] (2011) Smartphone Wikipedia. Dostopno na: http://en.wikipedia.org/wiki/Smartphone

[4] (2011) Android software development Wikipedia. Dostopno na: http://en.wikipedia.org/wiki/Android_software_development

[5] W. Frank Ableson, Robi Sen, Chris King. »Android in action second edition«, 2011

[6] (2011) Eclipce (software) Wikipedia. Dostopno na: http://en.wikipedia.org/wiki/Eclipse (software).

[7] (2011) Java (programming language) Wikipedia. Dostopno na: http://en.wikipedia.org/wiki/Java (programming language)

[8] (2011) SQLite Wikipedia. Dostopno na: http://en.wikipedia.org/wiki/SQLite

[9] (2012) Android Developers. Dostopno na: http://developer.android.com/index.html