Primjena GIS-a u penjanju

Brečević, Samuel

Undergraduate thesis / Završni rad

2015

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Polytechnic Pula - College of Applied Sciences / Politehnika Pula - Visoka tehničko-poslovna škola s pravom javnosti**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:212:357895

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-26**

Image not found or type unknown

Repository / Repozitorij:

Digital repository of Istrian University of applied sciences

zir.nsk.hr

Image not found or type unknown

POLITEHNIKA PULA

Visoka tehničko-poslovna škola

Samuel Brečević

PRIMJENA GIS-A U PENJANJU

Završni rad

Pula, rujan 2015.

Završni rad preddiplomskog stručnog studija Politehnike

PRIMJENA GIS-A U PENJANJU

Student: Samuel Brečević
Studijski program: studij Politehnike
Smjer: Inženjerstvo proizvodnje
Mentor: dr. sc. Branimir Ružojčić, mr. sc.
Komentor: dr. sc. Boris Blagonić, dipl. ing. geodezije

Pula, rujan 2015.

SAŽETAK

Završni rad bavi se projektiranjem i implementiranjem GIS-a u aplikaciju za pametne mobilne telefone. Opisan je proces sustava za izradu aplikacije, principi rada pametnih telefona i geoinformacijskog sustava te su na temelju proučene i analizirane literature opisani postupci koje treba odraditi kako bi se projekt realizirao.

Prvi dio završnog rada sastoji se od uvoda u rad, u drugom dijelu opisuju se pametni telefoni i Android operacijski sustavi, u trećem dijelu opisuju se GPS odašiljači i prijamnici, programski jezici i baze podataka, dok je konkretnija procedura izrade aplikacije za primjenu GIS-a u penjanju opisana u četvrtom dijelu završnog rada.

Ključne riječi: geoinformacijski sustavi, „pametni“ telefoni, Android operacijski sustavi, baze podataka, programski jezici, GPS, GNSS.

SUMMARY

The thesis deals with projecting and implementing GIS in an application for smartphones. It describes the process of creating application, work principles of smartphones, geographical information systems, and based on researched and analyzed literature are described steps that must be taken to carry out the project.

First part of thesis is made of introduction into thesis, second describes smartphones and Android operating systems, in third part are described GPS transmitters and receivers, programming languages, databases, while specific procedure for creating application for implementing GIS into climbing is described in forth part of thesis.

Keywords: geographical information systems, smartphones, Android operating systems, databases, programming languages, GPS, GNSS.

SADRŽAJ

1. UVOD....6
1.1 Opis problema....6
1.2 Cilj i svrha....6
1.3 Polazna hipoteza....6
1.4 Metode istraživanja....7
1.5 Struktura završnog rada....7
2. PAMETNI MOBILNI UREĐAJI....8
2.1 Razvoj pametnih mobilnih uređaja....8
2.2 Android operacijski sustav....13
2.3 Povijest Android operacijskog sustava....13
2.3.1.1 Android platforma....14
2.3.1.2 Zastupljenost Android distribucija....16
3. OSNOVE GEOINFORMACIJSKOG SUSTAVA....18
3.1 Definicija GIS-a....18
3.2 Povijesni razvoj GIS-a....18
3.3 Komponente GIS-a....19
3.3.1 Hardver....20
3.3.2 Softver....21
3.3.2.1 Softver za obradu slika....21
3.3.2.2 CAD Softver....23
3.3.2.3 Softver za upravljanje bazama podataka....23
3.3.2.4 GIS softver....24
3.3.3 Podatci....25
3.3.4 Korisnici....25
4. PROCEDURA IZRADE APLIKACIJE ZA PAMETNE MOBILNE UREĐAJE....26
4.1 Programski jezici....26
4.2 Baze podataka....30
4.3 Odabir komponenata....32
5. ZAKLJUČAK....49
6. POPIS LITERATURE....51
7. POPIS SLIKA I GRAFIKONA....52

1. UVOD

U završnom radu bavit ćemo se istraživanjem dostupnih tehnologija za izradu Android aplikacije za pametne telefone. Da bismo mogli govoriti o Android aplikaciji potrebno je opisati operacijski sustav Android, povijest razvoja mobilnih operacijskih sustava koji su utjecali na razvoj Android operacijskog sustava te reći ponešto o samoj tehnologiji, odnosno hardveru s kojim je Android kompatibilan, i što on omogućava korisniku.

Potrebno je napomenuti kako sve više ljudi počinje koristiti pametne telefone, bez obzira na njihove demografske značajke. Brojne mogućnosti pametnih telefona dovele su do nezaustavljivog rasta u razvoju aplikacija za mobilne uređaje, a tržište aplikacija postalo je jedno od najbrže rastućih grana industrije. Ponuda aplikacija iz dana u dan sve je veća i moguće je naći aplikaciju za gotovo sve primjene: zabavu, posao, vijesti, hobije, komunikaciju, obrazovanje itd. Iako je ponuda aplikacija na tržištu jako velika, još ima mjesta za neke nove ideje, koncepte i principe koji još nisu zaživjeli.

Jedan dio tržišta pametnih telefona temelji se na Android operacijskom sustavu, dok se onaj drugi temelji na iOS-u. U završnom radu nećemo pridavati značenje iOS-u, već Androidu koji je baziran na Linux jezgri (eng. kernel) i otvorenog je koda. Današnji mobilni telefoni sve više poprimaju ulogu računala koje korisnici veći dio vremena drže uz sebe. Njihov napredak omogućio je ugradnju različitih senzora koji prikupljaju podatke o okružju.

Stoga svaki novi senzor ili izmjena u odnosu na klasično osobno računalo omogućuje kreiranje novih programa i aplikacija koji koriste iste senzore s ciljem unaprijeđenja tehnologije i pružanja usluge korisniku.

1.1 Opis problema

Penjačima iz stranih zemalja teško je pronaći lokaciju penjališta samo uz pomoć penjačkog vodiča i klasičnih metoda opisivanja lokacije penjališta. Problem je stranim penjačima i pronalaženje pojedinih sektora u samom penjalištu, a time i smjerova u tim sektorima.

1.2 Cilj i svrha

Cilj je pronaći rješenje i osmisliti projekt lakšeg lociranja penjališta te snalaženja na samom penjalištu ponajprije stranim penjačima, a potom i penjačima hobistima.

1.3 Polazna hipoteza

Uz pravilnu analizu problema, bit će moguće kreirati projekt kojim bi se olakšalo snalaženje penjača na penjalištima na području Istre.

1.4 Metode istraživanja

U izradi završnog rada korištene su sljedeće metode:

- opisna
- metoda analize i sinteze
- grafička.

1.5 Struktura završnog rada

Rad je strukturiran u 5 glavnih poglavlja: u prvom dijelu nalazi se uvod kojim se čitatelja uvodi u temu završnoga rada s opisom problema, ciljem i svrhom rada, polaznom hipotezom i metodama istraživanja. U drugom dijelu obrađena je tema pametnih mobilnih uređaja koja je strukturirana od povijesnog pregleda pametnih mobilnih uređaja, Android operacijskog sustava i kratkog povijesnog pregleda Android operacijskog sustava te njegove zastupljenosti na tržištu. Osnove geoinformacijskog sustava nalaze se u trećem dijelu završnog rada koji je također podijeljen na definiciju GIS-a, povijesni razvoj GIS-a i komponente koje se nadalje dijele na hardver, softver, korisnike i podatke. U četvrtome je dijelu, u poglavlju Procedura za izradu aplikacije za pametne mobilne uređaje, obrađen konkretan primjer izrade aplikacije pomoću GIS-a u penjanju te je sadržaj razdvojen na programske jezike, baze podataka i odabrane komponente. Zaključak završnog rada napisan je u petom dijelu rada, te nakon njega slijede popis slika i literature čime završni rad završava.

2. PAMETNI MOBILNI UREĐAJI

Pametni mobilni uređaji (telefoni) u današnje su vrijeme mnogo više nego telefoni, mnogo više od toga što se nekad mislilo da bi telefon trebao biti. Oni imaju mnogo više mogućnosti od samog slanja SMS (eng. Short Message Service) poruka te primanja i upućivanja poziva.

Pametni mobilni uređaji danas predstavljaju malo prijenosno računalo koje će bez većih poteškoća obavljati matematičke operacije, obrađivati grafiku, spajati se na internet i umrežavati u kućnu mrežu brže od nekih prosječnih kućnih računala kupljenih prije nekoliko godina. Dapače, oni danas imaju mnogo više senzora, veće ekrane, a tržište njihovim aplikacijama u nezaustavljivom je rastu.

2.1 Razvoj pametnih mobilnih uređaja

Prvi uređaji koji su kombinirali telefoniju i koji su odrađivali operacije bili su osmišljeni 1971. godine, kada je Theodore G. Paraskevakos napravio prvi koncept takvog uređaja. Godine 1974. on je i patentirao taj uređaj, da bi 1993. krenula i prodaja uređaja. On je prvi predstavio koncept inteligencije, procesiranja podataka i vizualnog prikazivanja na telefonima. Godine 1971. radio je za Boing u Huntsvilleu u Alabami gdje je i predstavio prvi odašiljač i prijamnik koji je mogao komunicirati daljinski, no nije imao neku svrhu sve do PDA-a (eng. Personal Data Assistant, hr. dlanovnik) koji je predstavljao malo ručno prijenosno računalo s mogućnostima rokovnika, adresara, podsjetnika i kalkulatora.

Prvi mobilni telefon koji je imao funkcije dlanovnika osmišljen je u IBM-u te je predstavljen 1992. godine na COMDEX-u, sajmu računalne industrije. Godine 1994. mogao se kupiti prvi takav uređaj BellSoutha pod imenom Simon Personal Communicator, ujedno i prvi uređaj koji se mogao nazvati „pametnim“ (Slika 2.1).

Slika 2.1: Simon Personal Communicator

Izvor:
http://en.wikipedia.org/wiki/Smartphone#mediaviewer/File:IBM_Simon_Personal_Communicator.png (06.03.2015.)

U ranim su 90-ima dlanovnici, ondašnji pametni mobilni uređaji, imali Palm OS, BlackBerry OS ili Windows CE operacijski sustav, a kasnije su se ti operacijski sustavi razvili u mobilne operacijske sustave. Godine 1996. ondašnja finska tvrtka Nokia proizvela je model Nokia 9000 koji je kombinirao dlanovnik koji je radio na GEOS V3.0 tvrtke Geoworks, operacijskom sustavu s mobilnim telefonom Nokia 2110. Ta dva modela spojena u jedan dala su prvi model prijeklopnog mobilnog pametnog telefona – na modelu je jasno vidljivo kako je gornji dio prijeklopnog mobilnog telefona preuzet od dlanovnika, dok je donji dio tipkovnica koja sliči računalnoj tipkovnici (Slika 2.2).

Slika 2.2: Model Nokia 9000

Izvor: http://dozadeblog.ro/wp-content/uploads/2014/02/Nokia-9000-Communicator.jpg (06.03.2015.)

Kao što je vidljivo na slici 2.2, dok je model Nokia 9000 preklopljen izgleda poput ondašnjih mobilnih telefona, no otklopljen jasno ukazuje na svoje mogućnosti koje su bile mnogo veće od ondašnjih mobilnih telefona – od same vanjštine, zaslona veličine 4.5", QWERTY tipkovnice koja je bila mnogo sličnija ondašnjim računalima nego mobilnim telefonima, do njegovih internih mogućnosti koje nisu vidljive na prvi pogled: Intelov procesor 24 MHz i386 i 8 MB (Mega byets – 10^9 byets) memorije, od toga 4 MB radne memorije, 2 MB programske memorije i 2 MB za korisničke podatke. Taj model imao je mogućnosti raznih aplikacija: BCalc – kalkulator, Terminal 9000 – program upravljanja serijski spojenim uređajima, SMS Chat – aplikacija za komunikaciju porukama (među istim uređajima), VNC Client – omogućava prikaz ekrana s VNC Servera, tekstualni način pregledavanja interneta, slanje faksa, Call-Log Manager – popis poziva, pa čak i neke igre.

Godine 2000. Ericsson Mobile Communications napravio je Ericsson R380, prvi uređaj koji je na tržištu slovio kao pametni telefon, a također je omogućavao funkcije mobilnog telefona i dlanovnika uz ograničeno pregledavanje interneta na ekranu osjetljivom na dodir; pokretao ga je Symbian operacijski sustav koji se razvio iz EPOC-a.

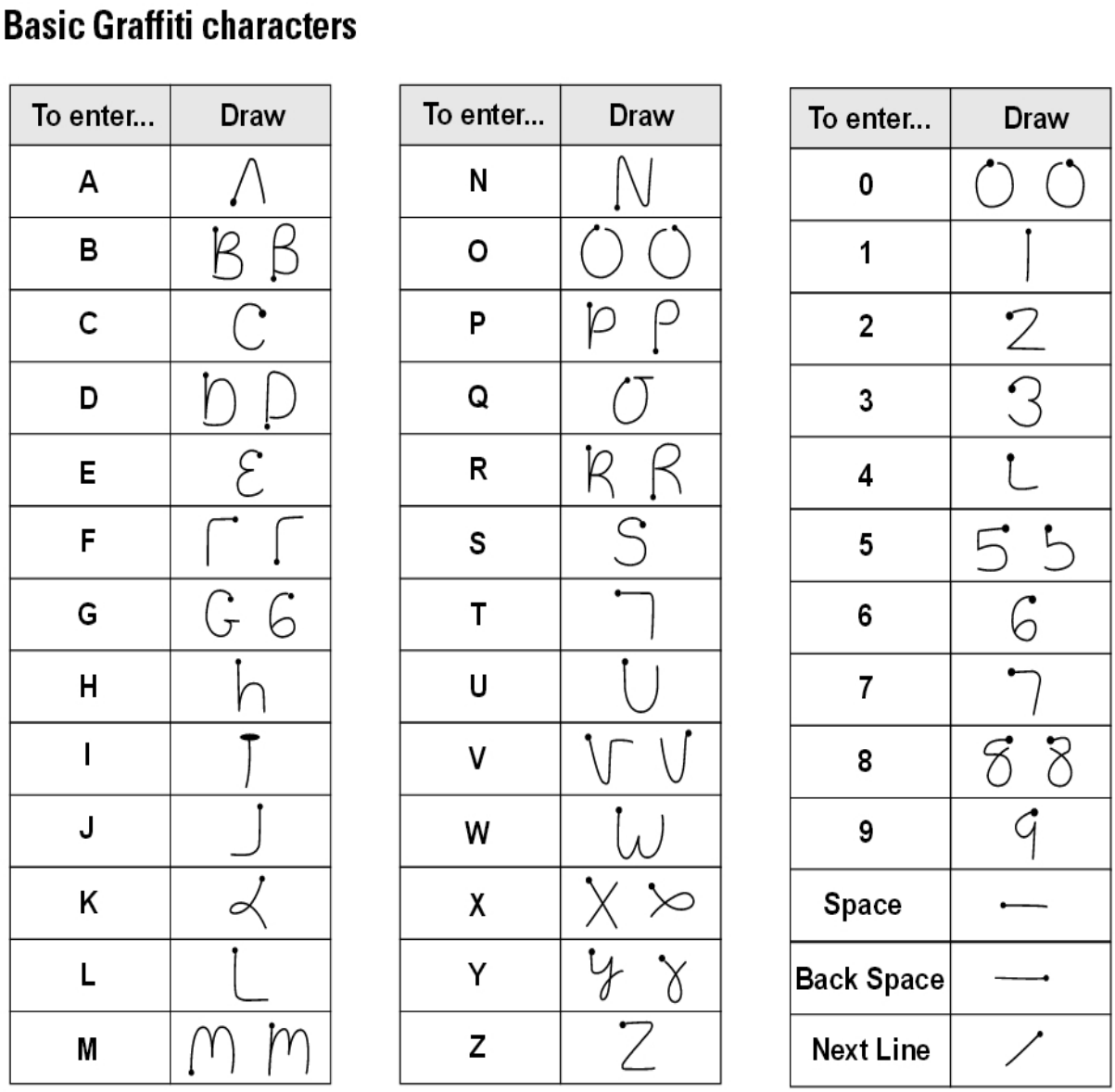
Početkom 2001. godine tvrtka Palm predstavila je prvi pametni mobilni telefon Kyocera 6035 koji je radio na Palm operacijskom sustavu (znan i kao Garnet), a također je obavljao funkcije mobilne telefonije, kao i funkcije dlanovnika. No za razliku od starijih modela mogao je odvojeno obavljati funkcije dlanovnika ili mobilnog telefona bez ovisnosti jednog softvera o drugom. Taj model također je imao obilježja prijeklopne tipkovnice, 20 MHz-ni procesor, 8 MB memorije te je mogao komunicirati putem infracrvenog senzora, a početna mu je cijena bila 499 $ u slučaju kupnje od operatera ili 549 $ u slučaju da je telefon kupljen bez ugovornih obveza (Slika 2.3).

Slika 2.3: Model Kyocera 6035

Izvor: http://en.wikipedia.org/wiki/Kyocera_6035#mediaviewer/File:Kyocera6035.jpg (07.03.2015.)

Takav model imao je zamjenjivu litijsku bateriju, svjetiljku sa stražnje strane uređaja te je omogućavao i punjenje uređaja pomoću RS232 kabela spajanjem baze na osobno računalo u COM port i, ono što je danas vrlo poznato, *gesture recognition* koje se u ono vrijeme zvalo *graffiti writing* – aplikacija koja je prepoznavala osnovne krivulje oblika slova koje bi uređaj „prepisao“ u latinično pismo (Slika 2.4).

Slika 2.4: Graffiti writing



Izvor: Uputstva za korištenje uređaja Kyocera 6035 (07.03.2015.)

2.2 Android operacijski sustav

Android je operacijski sustav namijenjen pametnim mobilnim telefonima, prvenstveno uređajima sposobnim prepoznati dodir na zaslonu. Sama jezgra operacijskog sustava bazirana je na Linux jezgri, a trenutačno ga razvija Google Inc., tvrtke iz Mountain Viewa, Kalifornija.

2.3 Povijest Android operacijskog sustava

Tvrtku Android Inc. osnovali su u Palo Altu, u Kaliforniji, u listopadu 2003. godine Andy Rubin, Rich Miner, Nick Sears i Chris White, da bi razvili, kako je Rubin izjavio, „pametnije mobilne uređaje koji su svjesniji vlasnikove lokacije i želja“. Rani izum bio je razvitak operacijskog sustava za digitalne kamere i fotoaparate, a s obzirom na to da tržište nije odgovaralo mogućnostima tvrtke odlučili su se na malo veće tržište, odnosno razvitak operacijskog sustava za pametne mobilne uređaje koji će konkurirati Symbianu i Microsoft Windows Mobileu. Vizija nakon operacijskog sustava bila je napraviti mobilni operacijski sustav koji će korisniku omogućiti pristup internetu te unaprijediti korištenje pametnog mobilnog telefona uz saznavanje lokacije i mobilnosti korisnika uređaja.

Godine 2005. Google Inc. kupuje Android Inc. kojemu je tadašnja ideja bila razviti operacijski sustav koji neće biti vezan za proizvođača hardvera, već sustav koji će svatko moći licencirati i instalirati na njihov uređaj kako bi on bio fleksibilan i održiv, i koji će biti izravna konkurencija Appleovom iPhoneu pod nazivom gPhone.

Godine 2007. Google Inc. oformio je Open Handset Alliance (OHA), udruženje 34 tvrtke (ponuđača mobilnih usluga i proizvođača hardvera) s Google Inc. i razvio prvi dostupan Android Software Development Kit (SDK) – skup alata za razvijanje mobilnih aplikacija za Android operacijski sustav u Java programskom jeziku koji je bio dostupan javnosti.

Godine 2008. OHA je razvila novu inačicu SDK-a pod Apache licencijom otvorenog koda te je objavila prvi spremni uređaj koji pokreće Android operacijski sustav; radilo se o telefonu znanom pod nazivima: T-mobile G1, HTC Dream ili Google G1. Taj je uređaj imao kapacitivni zaslon osjetljiv na dodir i potpunu QWERTY tipkovnicu, 528 MHz-ni procesor 256 MB ROM (Read Only Memory) memorije, 192 MB RAM (Random-access memory) memorije te proširive memorije do 16 GB (Giga bytes – 10^{12} bytes), 3,15 megapikselnu kameru (3 150 000 točkica na izvornu veličinu ekrana), a spajao se bežično Wi-Fi-em (Wireless Fidelity), Bluetoothom i žično putem EXT-USB kabela. Za razliku od prijašnjih pametnih mobilnih uređaja ovaj uređaj imao je A-GPS (Assisted-Global Positioning System), što je na hrvatskome jeziku asistirano globalno pozicioniranje, koji je definirao lokaciju ne samo pomoću satelita, već i pomoću odašiljača mobilnih davatelja usluga koji

je znatno ubrzavao pozicioniranje i lokaciju korisnika (Slika 2.5).

Slika 2.5: Google G1

Izvor: http://cdn2.gsmarena.com/vv/pics/t-mobile/t-mobile-g1-black.jpg (09.03.2015.)

Pojava Android telefona na tržištu krenula je sporo, no u 2009. godini se stabilizirala. Trenutačno Android koristi više od 7 svjetski poznatih proizvođača pametnih mobilnih telefona, kao npr.: Motorola, Sony Ericsson, LG, HTC i Samsung. Početkom siječnja 2010. godine Google Inc. objavio je kako je ime gPhone promijenjeno u Nexus One. Taj Googleov uređaj rađen je u tajvanskoj tvrtci HTC, koja je također napravila model HTC Dream, a to je isti uređaj drugog branda.

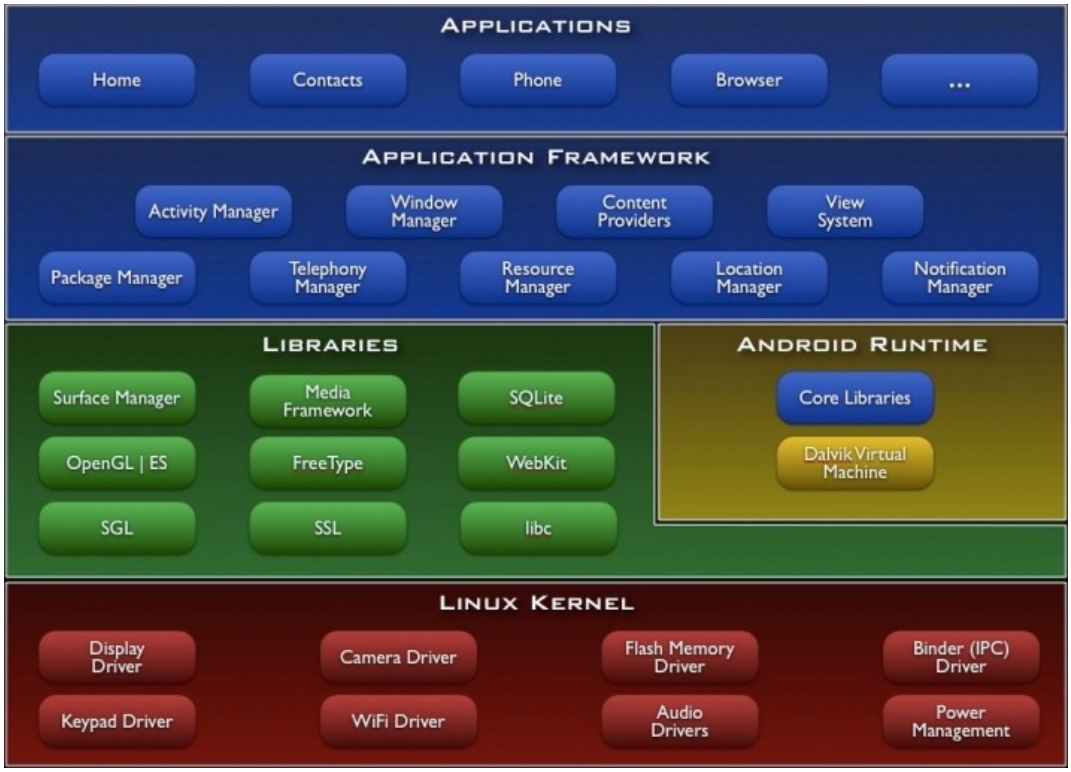
2.3.1.1 Android platforma

Android je više nego ime operacijskog sustava koji radi na Android pametnim mobilnim telefonima. Android se sastoji od nekoliko dobro definiranih slojeva razgranatih od Linux jezgre do ugrađenih aplikacija koje dolaze s određenim verzijama Android operacijskog sustava. Zasad je Android moguće instalirati na uređaje kojima se platforma temelji na ARM EABI, Intel x86 ili MIPS procesorskoj arhitekturi, a teoretski se može prenijeti na bilo koju hardversku platformu koja je podržana Linuxom.

Android operacijski sustav temelji se na Linux jezgri. Android verzije 1.0 temelji se na Linux jezgri verzije 2.6.27 koja nije dostupna u normalnoj Linux jezgri jer je mnogo mijenjana

kako bi se prilagodila novom hardverskom okružju. Modificirana Linux jezgra sadrži apstraktne hardverske promjene i upravljačke programe, kao što su upravljački programi za upravljanje zaslonom, kamerom, internetskim modemom, kao i upravljanje potrošnjom energije i sigurnostima. Programeri aplikacija za Android uređaje nikada se ne dotiču tog dijela softvera, već je to u nadležnosti proizvođača hardvera. Android također sadrži niz biblioteka koje upotrebljavaju različite komponente ove platforme, uključujući biblioteke poput libc, SQLite kao bazu podataka, FreType za renderiranje fontova, WebKit za HTML renderiranje, a također i biblioteke za reprodukciju medijskog sadržaja poput audio zapisa i crtanja 2D i 3D animacija. Android Runtime sastavljen je od dva dijela: SDK-a Virtual machinea koji pokreće aplikacije pisane sa SDK-om. SDK je dostupan programerima aplikacija za Android uređaje (3rd party developer) kao grupu Java okvira za rad s programskim jezikom i klasa pisanih u tom programskom jeziku. Virtual machine, zvana još i Dalvik, u Android pametnim mobilnim uređajima razlikuje se u tome što ne pokreće Java bytecode (kompajlirani Java source kod koji je ovisan o uređaju na kojemu se pokreće stoga i o Virtual machineu) izravno, već preko Dalvika kompajlira source kod. Kompajlirani kod nalazi se u .dex formatu datoteke i optimiziran je da radi na Dalvikovom virtualnom računalu te pokreće svaku aplikaciju kao novu instancu virtualnog računala.

Slika 2.6: Shematski prikaz arhitekture Android platforme



Izvor: http://elinux.org/images/c/c2/Android-system-architecture.jpg (13.03.2015.)

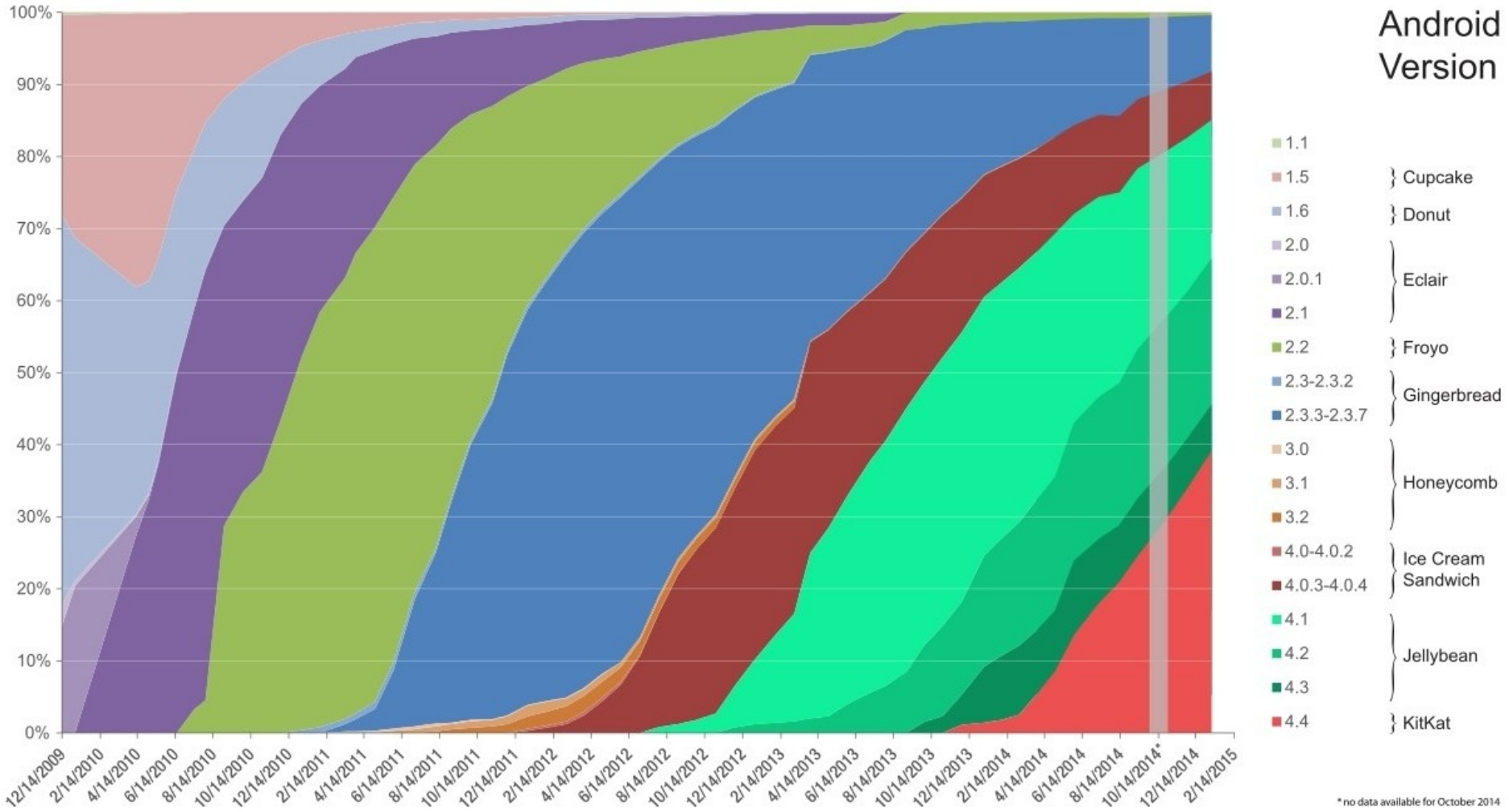
2.3.1.2 Zastupljenost Android distribucija

Od rujna 2008. godine pa sve do dana pisanja ovog završnog rada (ožujak 2014. godine) programeri su razvili 22 verzije Android operacijskog sustava, odnosno nadograđivali su od API (eng. application programming interface) verzije 1 do API verzije 22. API je skraćenica seta rutina, protokola i alata za kreiranje programa i alata. Od tih 22 API-ja, 12-ero njih imaju svoja imena, a ostali s istim imenima podverzije su Android operacijskog sustava, ali druge vrste API-ja:

1. Android 1.0 (API verzija 1) zove se Alpha
2. Android 1.1 (API verzija 2) zove se Beta
3. Android 1.5 (API verzija 3) zove se Cupcake
4. Android 1.6 (API level 4) zove se Donut
5. Android 2.0 (API level 5) zove se Éclair
6. Android 2.0.1 (API level 6) zove se Éclair
7. Android 2.1 (API level 7) zove se Éclair
8. Android 2.2–2.2.3 (API level 8) zove se Froyo
9. Android 2.3–2.3.2 (API level 9) zove se Gingerbread
10. Android 2.3.3–2.3.7 (API level 10) zove se Gingerbread
11. Android 3.0 (API level 11) zove se Honeycomb
12. Android 3.1 (API level 12) zove se Honeycomb
13. Android 3.2 (API level 13) zove se Honeycomb
14. Android 4.0–4.0.2 (API level 14) zove se Ice Cream Sandwich
15. Android 4.0.3–4.0.4 (API level 15) zove se Ice Cream Sandwich
16. Android 4.1 (API level 16) zove se Jelly Bean
17. Android 4.2 (API level 17) zove se Jelly Bean
18. Android 4.3 (API level 18) zove se Jelly Bean
19. Android 4.4 (API level 19) zove se KitKat
20. Android 4.4 (API level 20) zove se KitKat with wearable extensions
21. Android 5.0–5.0.2 (API level 21) zove se Lollipop
22. Android 5.1 (API level 22) zove se Lollipop

Grafički prikaz korištenja Android operacijskog sustava i pojedinih distribucija nalazi se na slici 2.7.

Slika 2.7: Grafički prikaz dostupnosti pojedinih API-ja kroz vrijeme



Izvor: http://upload.wikimedia.org/wikipedia/commons/e/ee/Android_historical_version_distribution_-_vector.svg (17.03.2015.)

3. OSNOVE GEOINFORMACIJSKOG SUSTAVA

3.1 Definicija GIS-a

GIS (geoinformacijski sustav, eng. geographic information system) relativno je nov pojam. Pojavio se kada i ostali informacijski sustavi, tj. pojavom računala. Općenito, sustav je skup povezanih objekata i aktivnosti koji svojim međuodnosima služe zajedničkoj namjeni. U GIS-u zajednička je namjena donošenje odluka pri upravljanju nekim prostornim aktivnostima. Informacijski sustav skup je postupaka izvršenih nad skupom podataka kojima se dobiva informacija pogodna za donošenje odluka. GIS možemo smatrati tehnologijom (hardver i softver) ili strategijom za obradu informacija, ovisno o kontekstu. Svrha je GIS-a unaprijediti donošenje odluka koje su na bilo koji način u vezi s prostorom.[1]

Sustav GIS čine zajedno:

- hardver
- softver
- podatci
- korisnici.

3.2 Povijesni razvoj GIS-a

Prvi koji se služio terminom geoinformacijskog sustava bio je Roger Tomlinson 1968. godine, koji je također prvi prihvaćen kao „otac GIS-a“. Godine 1832. prvi se put primjenjivala prostorna analiza radi epidemije kolere koja je zavladala Parizom, što je učinio Charles Picquet u analizi 48 četvrti Pariza. Godine 1854. John Snow također je analizirao teritorij zbog kolere, no to je bio slučaj u Londonu te prva uspješna zabilježena primjena geografske metodologije u epidemiologiji. Početkom 20. stoljeća zabilježen je razvoj u crtanju mapa, i to razvoj crtanja mapa u slojevima što je poboljšalo čitkost mapa jer su slojevi bili donekle neovisni jedni o drugima, što je projektantima olakšavalo posao, iako je i crtanje u slojevima bio vrlo zahtjevan posao. U početku su se slojevi crtali na staklu, što se promijenilo razvojem plastičnih filmova koji su bili mnogo lakši, manje krhki i praktičniji za transport. Kasnije su se ti slojevi pohranjivali pomoću velikih procesnih kamera od kojih bi nastala jedna slika za sve slojeve. To se još nije smatralo GIS-om jer nije bilo povezano s nikakvom bazom podataka, s obzirom na to da je bilo samo slika.

Potaknut istraživanjem nuklearnog oružja, razvoj računala vodio je razvoju programa koji bi služili za mapiranje te je 1960. godine razvijen prvi funkcionalni GIS u Ottawi, u Kanadi. Razvio ga

1 Dražen Tutić, Nada Vučetić, Miljenko Lapaine; *Uvod u GIS*

je u upravi šumarstva i ruralnog razvoja dr. Roger Tomlinson pod nazivom Canada Geographic Information System (CGIS). Taj Gis služio je za pohranjivanje, analiziranje i upravljanje podatcima koji su sadržavali informacije o zemlji, poljoprivredi, produktivnosti, životinjskom svijetu, vodenim tokovima, šumarstvu i zemlji u omjeru 1 : 50 000. CGIS bio je napredniji od dotadašnjeg mapiranja jer je omogućavao prekrivanja slojeva, mjerenja i digitaliziranja, odnosno skeniranja, a podržavao je i nacionalni koordinatni sustav koji se primjenjivao širom kontinenta. Godine 1990. CGIS imao je veliku bazu podataka o Kanadi i veliko računalo na kojem su se nalazili svi podatci. Njegova snaga bila je u analiziranju kompleksnih skupova podataka te nikad nije zaživio u komercijalne svrhe.

Godine 1964. Howard T. Fisher oformio je laboratorij za računalnu grafiku i prostornu analizu gdje je razvijeno mnogo važnih teoretskih koncepata u upravljanju prostornim podatcima koji su se u 1970-ima primjenjivali na softveru, kao što je SYMAP, GRID i ODYSSEY – to je služilo kao temelj za komercijanu upotrebu GIS-a na fakultetima, centrima za istraživanja i tvrtkama diljem svijeta. Kasnije u 70-ima bile su u razvoju i dvije javne domene GIS sustava Map Overlay and Statistical System (MOSS) i GRASS GIS, a u ranim 80-ima krenulo se u komercijalizaciju softvera koji je imao mnoge komponente CGIS-a, kao i novije načine upravljanja podatcima.

Godine 1986. razvija se Mapping Display and Analysis System (MIDAS), prvi GIS za komercijalnu primjenu na osobnim računalima pod Disk Operating Systemom (DOS). Razvojem interneta i operativnih sustava razvijao se i GIS te je tako u GIS uveden i Computer Aided Design (CAD), a ime MIDAS promijenjeno je u MaoInfo. Nakon tog GIS-a rapidno su se razvijali i drugi, podržavajući i druge operativne sustave te su prilagođavani za gotovo svaku vrstu hardvera.

3.3 Komponente GIS-a

S obzirom na to da je GIS sustav za prikupljanje, spremanje, provjeru, integraciju, upravljanje, analiziranje i prikaz podataka koji su prostorno povezani sa Zemljom, cjelokupni sustav mora biti sastavljen od komponenata kako bi funkcionirao.

3.3.1 Hardver

Najveća skupina hardvera u GIS-u jesu računala i ona su jedna od bitnijih komponenti GIS-a, a dijele se još i na:

- prijenosna
- osobna
- velika računala.

Prijenosna računala, zbog baterije u koju je pohranjena određena količina električne energije, u većini slučajeva služe za rad na terenu, radi omogućenog izravnog upisivanja i pohranjivanja podataka, a mogu služiti i kao osobna – stolna računala, jer u istom kućištu objedinjuju tipične komponente osobnog računala, uključujući zaslon, tipkovnicu, pokazivački uređaj i zvučnike.

Osobna su računala u masovnoj uporabi širom svijeta, a u usporedbi s prijenosnim računalima malo su jeftinija za iste performanse, ali im je mana to što su većinom vezana za jedno mjesto rada. Osobna i prijenosna računala izrađena su za rad jednog korisnika po svakom računalu, dok su velika računala namijenjena za rad i korištenje njihovih resursa od strane više korisnika istovremeno.

Velika se računala koriste u velikim organizacijama za izvršavanje velikih i zahtjevnih zadataka. Rade bez prestanka 24 sata na dan, 365 dana u godini i smještena su u klimatiziranim prostorijama s posebnim sustavima sigurnosti (od požara, nestanka struje itd.). Većinom služe kao serveri baza podataka, za obavljanje zahtjeva korisnika.

U hardver također spadaju i razni uređaji za mjerenje, pozicioniranje i digitalizaciju kao što su:

- GPS prijamnici
- totalne stanice
- sateliti
- digitalna fotogrametrijska kamera
- digitalni fotoaparat
- skeneri.

GPS prijamnikom moguće je odrediti položaj na površini i iznad nje bilo gdje na Zemlji, što je omogućeno sustavom posebnih satelita i uređaja na Zemlji. Razlikujemo precizne i ručne GPS prijamnike o kojima, između ostalog, ovisi i točnost određivanja položaja koja se može kretati od stotinjak metara do manje od 1 cm.

Totalne stanice posebni su uređaji za mjerenje koji omogućuju izmjeru terena geodetskim metodama. Omogućuju izmjeru i ispod površine Zemlje (u tunelima i sl.).

Postoje posebni sateliti s namjenom snimanja Zemljine povšine. Vrijednost satelitske snimke mjeri se rezolucijom koja danas već dostiže 1 m.

Digitalnom fotogrametrijskom kamerom slika terena dobiva se u digitalnom obliku neposredno čitljivom pomoću računala.

Skeneri digitaliziraju mape ili slike kako bi se pohranile u računalo. Postoje skeneri A4 formata koji su najčešće u primjeni, a ima i većih za pohranjivanje većih formata karata.

3.3.2 Softver

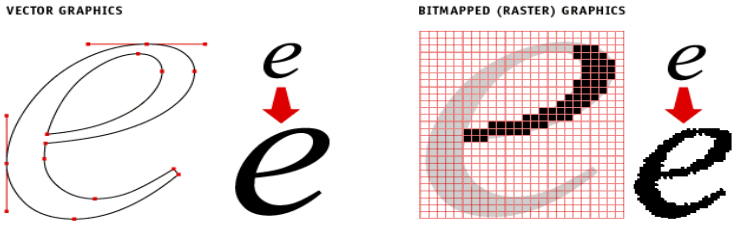
Spomenuti hardver, odnosno sva računala, pokreće neka vrsta Microsoftovog Windows operacijskog sustava ili sustava baziranog na UNIX operacijskom sustavu. U softver spadaju i programi i aplikacije koji služe za:

- obradu slike
- CAD (Computer-aided design; računalom podržani dizajn)
- programi za upravljanje bazama podataka
- GIS softver.

3.3.2.1 Softver za obradu slika

U softver za obradu slika spadaju programi i aplikacije za obradu rasterskih i vektorskih slika. Razlika između rasterske i vektorske slike jest ta što rasterska slika ima svoju rezoluciju, odnosno x puta y točaka (piksela), što rezultira lošijom slikom ukoliko se slika uvećava ili smanjuje, dok je kod vektorskih slika to nebitno, odnosno neovisne su o uvećanju ili umanjenju i kod njih je sve u linijama odnosno vektorima. Drugim riječima, vektorske i rasterske slike razlikuju se u tome što vektorska slika elemente slike opisuje geometrijskim oblicima i matematičkim formulama, a rasterska slika predstavljena je pomoću matrice točaka, zbog čega vektorske slike zauzimaju manje memorije od rasterskih (Slika 3.1).

Slika 3.1: Razlika između rasterske i vektorske slike



Izvor: http://apogeesigns.com/wp-content/uploads/2013/02/vector_raster.gif (08.04.2015.)

Neki od softvera za obradu rasterskih slika koji se naplaćuju:

- Adobe Photoshop
- ACD Canvas
- Corel PaintShop Pro i drugi.

Neki se softveri ne naplaćuju i moguće je s njima komercijalno raditi, a i otvorenog su koda:

- GIMP
- GNU Paint
- gThumb
- Tux Paint i mnogi drugi.

Neki od softvera za obradu vektorskih slika koji se naplaćuju:

- CorelDRAW
- Macromedia FreeHand
- Adobe Illustrator
- Microsoft Visio i drugi.

Neki se softveri ne naplaćuju i moguće je s njima komercijalno raditi:

- Dia
- Inkscape
- OpenOffice.org Draw/ LibreOffice Draw
- Synfig i mnogi drugi.

3.3.2.2 CAD Softver

CAD je skraćenica od Computer-Aided Design ili na hrvatskom jeziku računalom podržani dizajn, i označava uporabu računala kroz proces kreiranja i stvaranja dokumentacije. Korištenjem CAD programa povećava se produktivnost, kvaliteta dizajna i točnost proračuna, s obzirom na rad na projektima bez računala nekada u prošlosti.

Neki od CAD softvera koji se naplaćuju:

- ArchiCAD
- AutoCAD
- CATIA
- SolidWorks
- SketchUp i mnogi drugi.

Neki od CAD softvera koji se ne naplaćuju:

- LibreCAD
- OpenSCAD
- QCad
- SALOME i drugi.

3.3.2.3 Softver za upravljanje bazama podataka

SQL (eng. Structured Query Language) baze podataka organizirane su skupine podataka te da bismo ih održavali i upravljali njima potreban je softver. Softver za upravljanje i održavanje baza podataka u interakciji je s korisnikom, drugim programima i aplikacijama te samom bazom podataka.

Od softvera koji se naplaćuju mogu se izdvojiti:

- Microsoft SQL Server Management Studio
- Oracle Enterprise Manager
- SQL Database Studio
- Toad i drugi.

Softveri za upravljanje bazama podataka koji se ne naplaćuju jesu:

- MySQL Workbench
- Orbada
- pgAdmin III
- SQuirreL SQL i drugi.

3.3.2.4 GIS softver

GIS softver obuhvaća širok spektar programa i aplikacija koji uključuju kombinacije digitalnih karata i prostornih podataka, a može se svrstati u opće podjele:

- GIS softver otvorenog koda
- komercijalni GIS softver.

U GIS softver otvorenog koda spadaju mnogi programi i aplikacije od kojih su izdvojeni samo neki:

- QGIS
- GRASS GIS
- gvSIG
- ILWIS
- GeoServer
- Mapnik
- PostGIS
- GeoBase i mnogi drugi,

dok u komercijalne softvere spadaju:

- MapGuide
- ArcMap
- Oracle Spatial
- MicroStation i drugi.

3.3.3 Podatci

Podatci se smatraju i najvažnijom i najskupljom komponentom cjelokupnog GIS-a, a općenito se dijele na dva dijela:

- prostorne podatke
- opisne podatke.

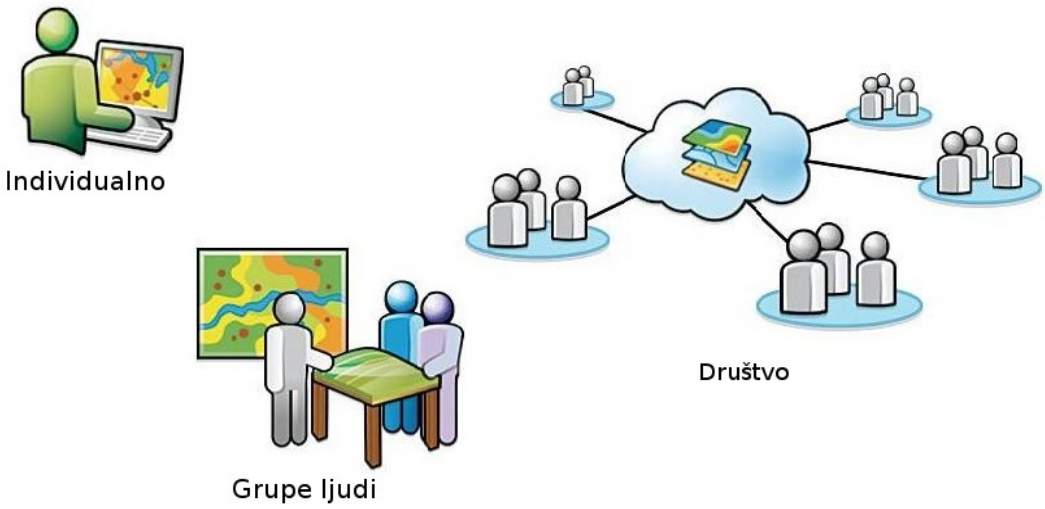
U prostorne podatke spadaju podatci koji su direktno vezani uz objekte na kartama GIS-a, kao što su adrese ucrtanih objekata, zgrade, ceste, željezničke pruge, stupovi rasvjete, vodovodi, električni vodovi i mnogi drugi. Dijele se na vektorske i rasterske, a odgovaraju na pitanja apsolutne i relativne lokacije ucrtanih objekata.

U opisne podatke spadaju podatci koji su indirektno vezani uz objekte na kartama GIS-a i odgovaraju na kvantitativna i kvalitativna pitanja, a mogu biti dijagrami, slike, tablice, financijski podatci, nazivi mjesta, imena, podatci o količinama, nadmorske visine. Opisni se podatci pretežito nalaze u bazama podataka i kvalificiraju kao nominalni, intervalni, redni i razmjerni.

3.3.4 Korisnici

Korisnici u GIS-u mogu biti pojedinci, a mogu biti i grupe ljudi u čijem slučaju korištenje GIS-a može biti u privatne i komercijalne svrhe, ovisno o licenciji softvera (Slika 3.2).

Slika 3.2: Korisnici GIS-a



Izvor: autor

4. PROCEDURA IZRADE APLIKACIJE ZA PAMETNE MOBILNE UREĐAJE

U proceduru izrade aplikacije spada odabir programskog jezika za izradu same aplikacije, odabir baze podataka odnosno aplikacije za održavanje baze podataka, u slučaju grafičkog sučelja odabir softverskog okvira za izradu, prikupljanje podataka i informacija te sjedinjavanje svega navedenoga u jedno.

4.1 Programski jezici

Programski je jezik striktno (prema pravilima) napisani jezik kreiran za komunikaciju između programera i računala kako bi računalo razumjelo instrukcije dane pritiskom tipki, određenim prelaskom pokazivača preko objekta, pokretom, dodirom ili već predodređenom interakcijom korisnika.

Svaki se programski jezik prevodi u mašinski ili strojni jezik koji je poseban za određeno računalo ili stroj, odnosno ovisi o arhitekturi računala. Viši se programski jezici na strojni prevode pomoću kompajlera ili se naredbe u višem programskom jeziku prevode u niži programski jezik, npr. Assembler.

Primjer prevoditeljskog (interpretiranog) programskog jezika bio bi Basic (izvršava se unutar posebnog programa tzv. prevoditelja (eng. interpreter) koji stoji između računala i programa). Postoje niži i viši programski jezici. Npr. Assembler ili strojni kod primjer je nižeg programskog jezika (obično se radi o izravnom pozivanju „naredbi centralne jedinice“ (CPU) ili instrukcija kao npr. INC (uvećanje), MOV (kopiranje spremnika (registara)) itd.). Primjer višeg programskog jezika je C koji se mora najprije prevesti, a potom prilagoditi da radi na određenom operacijskom sustavu (eng. compile & link). Postoje drugi oblici klasifikacija i tipovi programskih jezika te je vjerojatno važno spomenuti objektno-orijentirane jezike (C++, Java itd.) koji su danas najrašireniji u primjeni, koristeći razne mehanizme kakvi nisu bili u uporabi u npr. C-u i Pascalu. Najbitniji su pojmovi u tom slučaju objekt, enkapsulacija, nasljeđivanje itd. Pri programiranju također se koriste razne metodologije razvoja softvera (programske podrške) pri kojima je dobro početi od vodopadnog modela, spiralnog, prototypinga i sličnih, kao jednostavnih modela razvoja. Standardi za razvoj i osiguranje kvalitete (kakvoće) softvera također postoje te je bitno spomenuti ISO standarde. Veće tvrtke koje se bave razvojem softvera obično koriste svoje metodologije razvijane godinama ili prilagođavaju postojeće kako bi pratile njihovu filozofiju razvoja, a sve u

svrhu osiguranja kvalitete i uniformnosti razvoja.[2]

Prvi programski jezik koji je bio upotrebljiv na elektroničkom računalu, a ne na mehaničkim strojevima, bio je Short Code koji je razvio John Mauchly 1949. godine. Naredbama u Short Codeu računalo je izvodilo matematičke operacije, no program je morao biti prevođen u mašinski kod svakog puta kada bi se pokrenuo produžavajući tako proces izvođenja programa za razliku od samog izvođenja mašinskog koda.

80-ih godina slijedilo je relativno sjedinjenje, pa je tako C++ (jezični „potomak“ C-a) kombinirao objektno-orijentirano i sistemsko programiranje, dok se u Japanu istraživala peta generacija programskih jezika koja se temeljila na logičkom programiranju. Bitan trend bio je oblikovati programski jezik u širokom spektru usredotočenog na module, a to su i radili Modula-2, Ada, i ML (metalanguage).

Brzi razvoj interneta sredinom 1990-ih pružao je prilike za razvoj novih programskih jezika među kojima jesu:

- Perl – kreiran 1987. godine i razvijen prvotno kao Unix alat za pisanje skripti, postao je glavnim korištenim jezikom za dinamične internetske stranice.

- Hashkell – kreiran 1990. godine, standardizirani programski jezik opće svrhe bez striktne shematike s jako statičkim pisanjem.

- Python – kreiran 1991. godine, široko korišteni viši programski jezik opće namjene čija je filozofija: čitljivost koda i programiranje u manje linija koda nego što je to kod Jave i C++-a. Također je vrlo jednostavan te se preporuča kao programski jezik za učenje u nekim obrazovnim ustanovama.

- Visual Basic – kreiran 1991. godine, programski jezik koji je potaknut događajem i u sebi sadržava IDE (eng. Integrated Development Environment) razvojno okruženje.

- PHP – kreiran 1995. godine i koristi se na serverima, kreiran prvenstveno za razvoj internetskih stranica, no koristi se i kao programski jezik opće namjene.

2 https://hr.wikipedia.org/wiki/Računalno_programiranje

- Ruby – kreiran 1995. godine, dinamički pisan, reflektivan, objektno orijentiran programski jezik opće namjene.

- Java – kreirana 1995. godine, upotrebljavana na serverima i bytekod virtualnim strojevima u komercijanim svrhama uz obećanje: „Jednom napisano, pokrenuto bilo gdje“. Ovaj je programski jezik zadržao popularnost i danas se uz C najviše koristi.

- JavaScript – kreiran 1995. godine, dinamički programski jezik najčešće korišten za interakciju s internetskim preglednikom, odnosno za programiranje internetskih stranica s korisničke strane.

Za izradu Android aplikacije izbor programskih jezika bitno se sužava jer sadašnji pametni mobilni uređaji u sebi sadrže hardver koji mora odgovarati softveru, te Android službeno podržava Javu kao službeni programski jezik.

Neki od mogućih jezika za programiranje aplikacija na Android pametnim mobilnim uređajima jesu:

- Basic4android – komercijalni proizvod sličan Slimpleu, inspiriran Visual Basicom i Visual Studiom. Čini programiranje u Androidu mnogo jednostavnijim od običnog Visual Basica programerima kojima se programiranje u Javi čini teškim te postoji velika skupina ljudi u svijetu koji programiraju u njemu i spremni su na pitanja i te problematike.

- Corona SDK – SDK koji dopušta razvijanje aplikacija za iPhone i Android pametne mobilne uređaje, ujedno podržava i razvijanje GUI (eng. Graphical User Interface) aplikacija u Lua programskom jeziku što je građeno na površini C++ i OpenGL-a (eng. Open Graphics Library) koji je API za mnoge platforme koje za GUI aplikacije koriste grafičku karticu.

- Delphi – oblik objektnog Pascala, također za programiranje Android aplikacija.

- HyperNext Android Creator – softver za razvoj Android aplikacija namijenjen programerima početnicima kojima nije potrebno znanje u Javi. Temeljen je na HyperCardu koji tretira softver kao snop karata od kojih je samo jedna vidljiva, stoga je pogodan za razvoj mobilnih aplikacija koje imaju vidljiv samo jedan prozor u datom trenutku. HyperNext Android

Creatorov glavni programski jezik naziva se još i HyperNext i ima sličnosti s Hypercardovim HyperTalk programskim jezikom. HyperNext je interpretacija engleskog jezika i podržava mnogo opcija za razvijanje Android aplikacija te također podržava rad aplikacije u pozadini što olakšava procesiranje podataka i sam povratak na aplikaciju nakon određene odsutnosti od procesa pokrenute aplikacije.

- Kivy – biblioteka otvorenog koda pisana u Pythonu za razvijanje multi-touch aplikacija s „Prirodnim korisničkim sučeljem“ (eng. Natural user interface – NUI), što predstavlja korisničko sučelje koje je efektivno nevidljivo i ostaje nevidljivo kako korisnik konstantno uči uvećavajuće kompleksne interakcije s uređajem; naziva se prirodnim jer su računalna sučelja najčešće programirana tako da čovjek mora naučiti koristiti softver i u tome nema mnogo slobode. Kivy također omogućava održavanje jedne aplikacije za brojne operacijske sustave na principu „jednom napisano, pokrenuto bilo gdje“, na:
    - Windows OS-u
    - Linux OS-u
    - Android OS-u
    - iOS-u.

Pomoću alata nazvanog Buildozer eksportira se aplikacija iz Linux OS-a na Android uređaje.

- Lazarus – upotrebljavan za razvoj aplikacija pomoću Pascal programskog jezika s Free Pascal kompajlerom počevši od verzije 2.7.1.
- Qt for Android – upotrebljava se za razvoj aplikacija na:
    - Android OS-u
    - Linux OS-u
    - Windows OS-u.

Razvoj je moguć s C++ i QML programskim jezikom i uz pomoć Android SDK i NDK alata.

- RFO BASIC! – dijalekt jezika Darthmouth Basic, interpreter na uređaju zajedno s bibliotekama za senzore, multi-touch, SQLite bazu podataka, HTML GUI, telefoniju i dr. Najvažnije je da je on otvorenog koda, no aktivan je samo od ožujka 2015.

- RubyMotion – pisanje u Rubyju, također podržava pozivanje API-ja iz Jave izravno u Ruby, te se statički kompajlira u mašinski kod.

- Saphir – sličan kao Kivy, ali pisan u Rebolu, uključujući grafičke mogućnosti, pristup mreži i datotekama na navedenim operacijskim sustavima kao kod Kivyja.

- SDL (Simple Directmedia Layer) – nudi programiranje isključujući Javu, dopuštajući C programski jezik.

- Xamarin – razvoj aplikacija u C# okruženju za distribuciju na iOS i Android uređaje, od veljače 2014. godine broji preko 505 000 programera.

- I mnogi drugi.

4.2 Baze podataka

Baze podataka organizirane su skupine podataka te da bismo ih održavali i upravljali njima potreban je softver. Softver za upravljanje i održavanje baza podataka u interakciji je s korisnikom, drugim programima i aplikacijama te samom bazom podataka.

Jedna od mogućih definicija baze podataka glasi da je to zbirka zapisa pohranjenih u računalu na sustavan način, takav da joj se računalni program može obratiti prilikom odgovaranja na problem. Svaki se zapis za bolji povratak i razvrstavanje obično prepoznaje kao skup elemenata (činjenica) podataka. Predmeti vraćeni u odgovoru na upitnike postaju informacije koje se mogu koristiti za stvaranje odluka koje bi inače mogle biti mnogo teže ili nemoguće za stvaranje. Računalni program korišten za upravljanje i ispitivanje baze podataka nazvan je sustavom upravljanja bazom podataka (SUBP). Svojstva i dizajn sustava baze podataka uključeni su u

proučavanje informatičke znanosti.[3]

Softveri se mogu podijeliti na komercijalne i nekomercijalne softvere za izradu i održavanje baza podataka, a neki su od poznatijih:

- MySQL
- PostgreSQL
- SQLite
- Microsoft SQL Server
- Oracle
- Sybase i mnogi drugi.

Softveri za izradu baze podataka također su programirani u nekom od sistemskih programskih jezika kao što su: C, Java, C++ ili neki drugi, no iako su ti softveri pisani u nekom od programskih jezika, za upravljanje i rad s bazama podataka i samim podatcima potrebno je poznavanje SQL jezika koji je univerzalan i standardiziran za sve DBMS-ove (eng. Database management systems). Sintaksa SQL jezika svodi se na značenje same riječi na engleskome jeziku:

- SELECT – odabir podataka iz baze podataka
- UPDATE – ažuriranje/dopuna ako podatak ne postoji u bazi podataka
- DELETE – brisanje podataka iz baze podataka
- INSERT INTO – ubacivanje novih podataka u bazu podataka
- CREATE DATABASE – kreiranje nove baze podataka

3 http://hr.wikipedia.org/wiki/Baza_podataka 09.06.2015.

- ALTER DATABASE – mijenjanje postojeće baze podataka
- CREATE TABLE – kreiranje nove tabele
- DROP TABLE – brisanje postojeće tabele iz baze podataka
- CREATE INDEX – kreiranje indeksa
- DROP INDEX – brisanje indeksa i mnoge druge.

Bitno je napomenuti da SQL jezik nije osjetljiv na velika i mala slova, iako je poželjno pisati ga velikim slovima radi bolje čitkosti u velikoj količini koda.

Primjer funkcionalne sintakse:

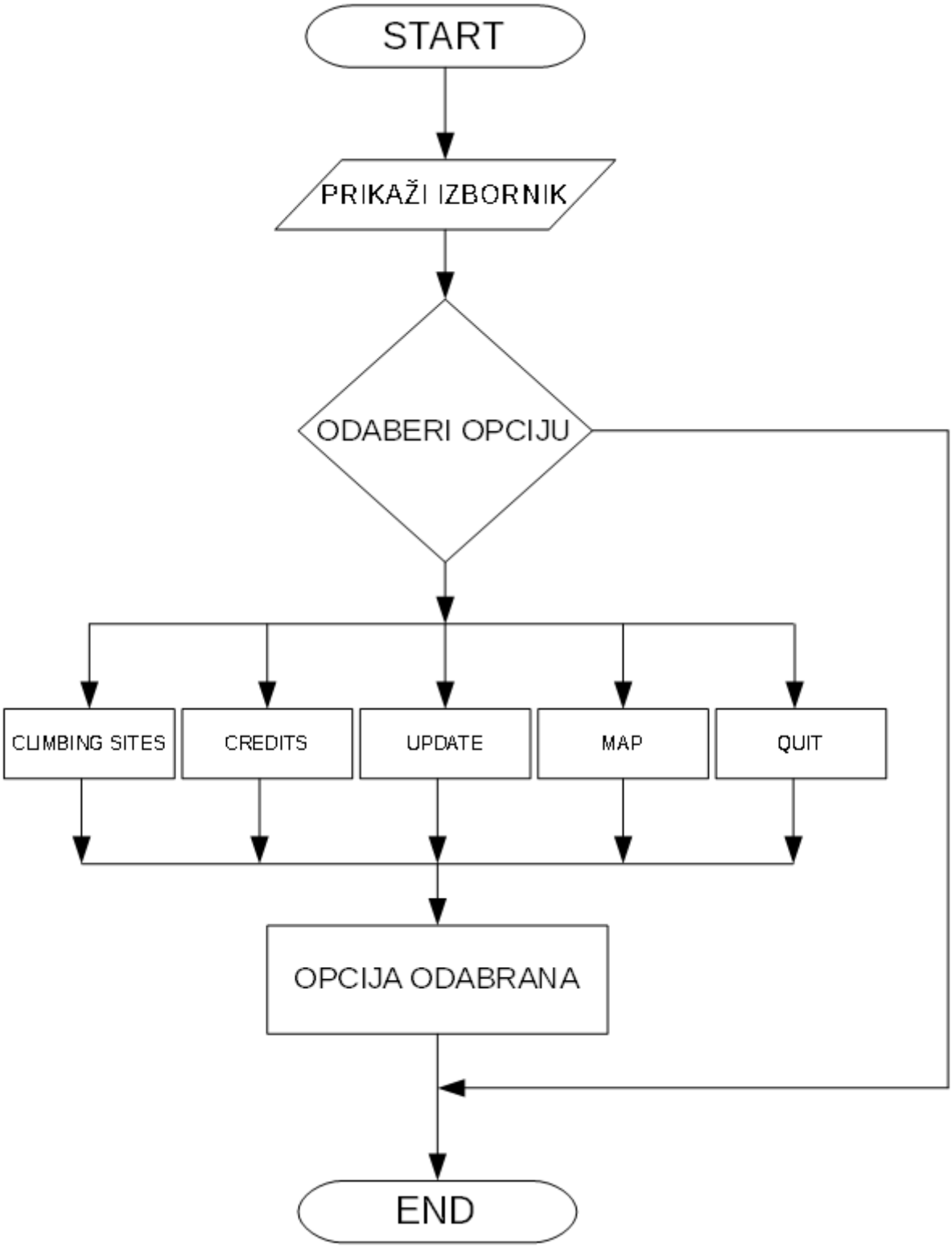
- SELECT * FROM penjalista WHERE duzina LIKE 25

Ovaj primjer u izvedbi bi označio (SELECT) sve podatke (*) iz (FROM) tabele imena „penjalista“ gdje (WHERE) je u stupcu imena „duzina“ podatak „sličan 25“.

4.3 Odabir komponenata

O kojim se god tehnologijama radilo, programskom jeziku, bazi podataka, potrebno je izraditi planski dijagram toka te teoretski dijagram toka, stoga bi za početni i glavni izbornik dijagram toka trebao izgledati kao na slici 4.1.

Slika 4.1: Dijagram toka za glavni izbornik



Izvor: autor

Pseudokod izgledao bi ovako:

Start aplikacije

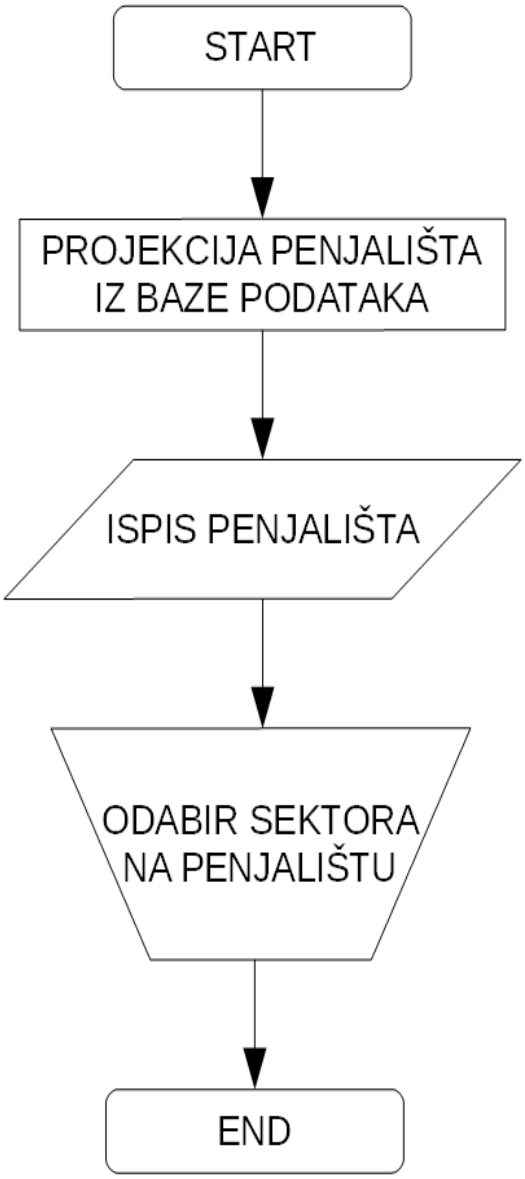
Prikaz izbornika

    Odabir opcije

Kraj

Dijagram toka za ekran Climbing sites vidljiv je na slici 4.2.

Slika 4.2: Dijagram toka za ekran Climbing sites



Izvor: autor

Pseudokod za ekran Climbing sites:

Projekcija penjališta iza baze podataka

    Ispis projekcije na ekran

Kraj

Sljedeći ekrani izgledaju jednako, samo je umjesto penjališta naveden sektor, a umjesto sektora smjer. No potrebno je napomenuti kako se pseudokod i dijagram toka ne razlikuju jedan od drugoga s obzirom na različiti odabir, ali ekrani se znatno razlikuju jer se upit u bazu podataka sastoji od odabrane tipke koja je na prvom ekranu penjalište te se s obzirom na penjalište ispisuju sektori samo u tom penjalištu i, naravno, smjerovi se u skladu s time ispisuju pomoću odabranog sektora, što znači da ima mnogo varijabilnih ispisa na ekran.

Od programskih jezika koji se pružaju na tržištu, poprilično jednostavan za učenje, kao i vrlo shvatljiv jezik, jest Python. Za razliku od Jave kod koje je nužno završavati zagradama i „točka-zarezima“, kod Pythona se to čini uvlačenjem koda za četiri razmaka ili jednim tabulatorom koda udesno. Python koristi uvlačenje kao metodu razlikovanja programskih blokova, tj. ne koristi vitičaste zagrade ili ključne riječi kao većina programskih jezika. Povećanje uvlačenja znači da dolazi novi, ugniježđeni blok, dok smanjenje označava kraj trenutačnog bloka.

U Pythonove ključne riječi spadaju:

- `if` izraz, koji izvršava određeni blok koda pod nekim uvjetom, zajedno s `else` i `elif` (else-if varijanta)

- `for` izraz, koji iterira kroz iterabilan objekt i svaki element upisuje u lokalnu varijablu koja se koristi u pridruženom bloku

- `while` izraz, koji izvršava određeni blok koda sve dok je njegov uvjet istinit

- `try` izraz, koji omogućava da iznimke, koje su bačene u bloku koda koji `try` obuhvaća, budu uhvaćene i obrađene u `except` bloku

- `class` izraz, kojim se deklarira klasa (u objektno orijentiranom programiranju)

- `def` izraz, koji definira funkciju ili metodu

- `assert` izraz, koji se koristi pri debugiranju kako bi se provjerilo vrijede li određeni uvjeti

- `import` izraz, koji se koristi pri uključivanju dodatnih modula.[4]

Također, kod Pythona nije potrebno definirati vrstu varijable što skraćuje vrijeme pisanja koda. Iako je u izvođenju programa Python lošiji od Jave, kod Pythona je pisanje samog koda od 3 do 5 puta kraće ovisno o programu, što je programeru i administratoru sustava najbitnije. Python je dinamički pisani jezik što znači da je moguće objektu mijenjati vrstu u bilo koju i bilo kada.

Pokretanje ovoga koda u Pythonu:

print "Hello world!" # za Python verzije <3, ili

print("Hello world!") # za Python verzije 3

izvodi se isto ako s Javom:

public class HelloWorld

{ public static void main (String[] args)

{ System.out.println("Hello World!");}

}

i to ispisuje Hello World! u komandnoj liniji.

Za definiranje klasa u Javi potrebna je također za svaku klasu po 1 datoteka, dok je u Pythonu moguće sve spremiti u jednu jedinu datoteku, iako nije poželjno zbog praktičnosti i čitljivosti koda.

Primjer Python koda za izradu aplikacije primjenom GIS-a u penjanju:

4 https://hr.wikipedia.org/wiki/Python_(programski_jezik) 18.06.2015.

Slika 4.3: Python kod u aplikaciji primjenom GIS-a u penjanju

```
class MainScreen(Screen):
    def citanje(self):
        global penjalista
        penjalista = []
        conn = sqlite3.connect('base.db')
        conn.text_factory = str
        c = conn.cursor()
        c.execute('SELECT penjaliste FROM plezanje')
        penjalista = c.fetchall()
        #izbacivanje istih penjalista iz lista
        counter = Counter(penjalista)
        penjalista = counter.keys()
        c.close()
        #SORTIRAM PENJALISTA PO ABECEDI
        penjalista.sort()
        penjalistaa = penjalista
        penjalista = []
        itr = 0
        while itr <= penjalistaa.index(penjalistaa[-1]):
            for penjaliste in penjalistaa:
                penjalista.append(penjaliste[0])
                itr = itr + 1
        penjalistaa = []
        return penjalista
    pass
```

Izvor: autor

U ovom bloku koda definiran je glavni ekran aplikacije, varijabla „penjalista“ je prazna lista, nakon koje se program spaja na bazu podataka, odabire stupac „penjalista“ iz tabele „plezanje“, prebrojava penjališta te sprema dobivena penjališta iz baze podataka u praznu listu „penjalista“. Lista se zatim pretvara u nepromjenjivu listu, nakon toga se nepromjenjiva lista vrti u petlji da bi se kreirala promjenjiva (neefikasan način, ali služi svrsi), nakon čega funkcija vrati punu listu s penjalištima.

Produkt tog dijela koda ne bi bio vidljiv jer nema naredbe za ispis na ekranu te program drži u memoriji varijablu penjališta.

Od navedenih baza podataka za Android aplikacije vrlo su zahvalne SQLite baze podataka s obzirom na to da nije potrebna aplikacija koja pokreće samu bazu podataka, već je baza podataka sadržana u datoteci kojoj se pristupa s lakoćom, a ono što je najbitnije, ona je open source, tj. sav programski kod SQLite baze zasnovan je na načelu javnog dobra. To znači da se programski kod u cijelosti i njegovi dijelovi slobodno mogu kopirati, modificirati, objavljivati, koristiti, prodavati,

distribuirati u izvornom SQLite formatu, otvorenom kodu ili u kompajliranom obliku za bilo koju komercijalnu ili nekomercijalnu svrhu.

SQLite relacijska je baza podataka temeljena na maloj C programskoj biblioteci. Glavne značajke baze podataka SQLite jesu:

- Transakcije funkcioniraju po načelu ACID (atomic, consistent, isolated, and durable). Podatci ostaju postojani pri padu sustava ili nestanku električne energije.
- Nulta konfiguracija nema potrebu za prilagođavanjem i administracijom.
- Podržava većinu SQL92 standarda.
- Cijela baza podataka smještena je u jednoj datoteci na disku.
- Podatci iz baze mogu se slobodno dijeliti između više računala.
- Podržava nekoliko terabajta velike baze i nekoliko gigabajta velikih nizova podataka.
- Brža izvedba standardnih operacija nego kod ostalih popularnih baza podataka. Rezultati testiranja.
- Jednostavno korisničko sučelje.[5]

5 https://hr.wikipedia.org/wiki/SQLite

Slika 4.4: SQLite baza podataka za izradu aplikacije primjenom GIS-a u penjanju

	route_id	penjaliste	sektor	ime_smj	ocjena_smj	duzina_smj	note
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
100	20621	Buzetski kanjon	Pengari C	Nevermind	7c+	32	
101	20622	Buzetski kanjon	Pengari C	Non habemus ...	7b+	30	
102	20623	Buzetski kanjon	Pengari C	Avanti forza	6b+	20	
103	20624	Buzetski kanjon	Pengari C	Avanti popolo	6a	35	
104	20625	Buzetski kanjon	Pengari C	Bolting weekend	6a	33	
105	20626	Buzetski kanjon	Pengari C	Silent is golden	6a	32	
106	20627	Buzetski kanjon	Pengari C	Pasareta	6a	33	
107	25067	Brseč	Belićev stup A	Goldfinger	6b+	30	
108	25068	Brseč	Belićev stup A	Octopussy	6a+	35	
109	25069	Brseč	Belićev stup A	Dr. No	6a+	40	
110	25070	Brseč	Belićev stup A	Casino Royale	6b+	40	
111	25071	Brseč	Belićev stup A	Moonraker	6a+	32	
112	25072	Brseč	Belićeva stijena B	The Ivy Plot	6b+	25	
113	25073	Brseč	Belićeva stijena B	Kitten in the need	6c+	25	
114	25074	Brseč	Belićeva stijena B	Wespennest	6b+	30	
115	25075	Brseč	Podgrad C	Meister Horich	5b	45	
116	25076	Brseč	Podgrad C	Diver in Danger	7a	50	
117	25077	Brseč	Podgrad C	Flying Dutchman	6a+	55	
118	25078	Brseč	Podgrad C	Crimson Hawk	6c+	50	
119	25079	Brseč	Podgrad C	III	6c	25	
120	25080	Brseč	Podgrad C	Wanderratz	5c+	50	
121	25081	Brseč	Podgrad C	Ratroof	6b+	20	
122	25082	Brseč	Podgrad C	Once upon a ti...	6a	50	
123	25083	Brseč	Podgrad C	Feige/Sau	6c+	50	
124	25084	Brseč	Podgrad C	Hope	6c	50	
125	25085	Brseč	Podgrad C	Mullmania	6c	30	

Izvor: autor

S obzirom na to da je kao programski jezik odabran Python, ne pruža se mnogo mogućnosti za odabir grafičkog sučelja na Android pametnim mobilnim uređajima, no postoji Kivy koji je također vrlo jednostavan za početnike te ima prirodno korisničko sučelje i podržava multi-touch. Ono što je najvažnije jest da je jednom napisani kod do verzije 1.8.0 moguće pokrenuti na Linux operacijskom sustavu, Windows operacijskom sustavu, iOS i Mac OS, Android operacijskom sustavu, a najpogodnije je razvijanje aplikacija za ekrane osjetljive na dodir.

Slika 4.5: Kivyjem podržani sustavi

Izvor: http://kivy.org/docs/_images/gs-introduction.png

Kivy nije kreiran iz nečega, odnosno ne nastavlja se na neki projekt, već kreće od početka – programeri su razvili takav jezik radi lakše komunikacije između računala i ljudi. Kivy je brz u razvoju aplikacija, kao i u izvođenju procesa zbog implementacije kritičnih funkcija na C razini kako bi dobio na snazi, te su pri programiranju korišteni inteligentni algoritmi za vremensku minimalizaciju operacija. Pisan je tako da koristi grafički procesor u slučajevima kada se smatra da znatno smanjuje vrijeme izvođenja koda, a budući da je grafički procesor u današnje vrijeme bolji od centralnog, to je i pogodnije za izvođenje grafičkih operacija. Možda najvažnije od svega jest da je Kivy besplatan i da u njegovoj izradi djeluju ljudi koji žive samo od njegovog razvoja, obučeni ljudi i profesionalci u svom području, te da taj projekt nije studentski hir, već je tu da bi ostao.

Slika 4.6: Prikaz koda u Kivy jeziku

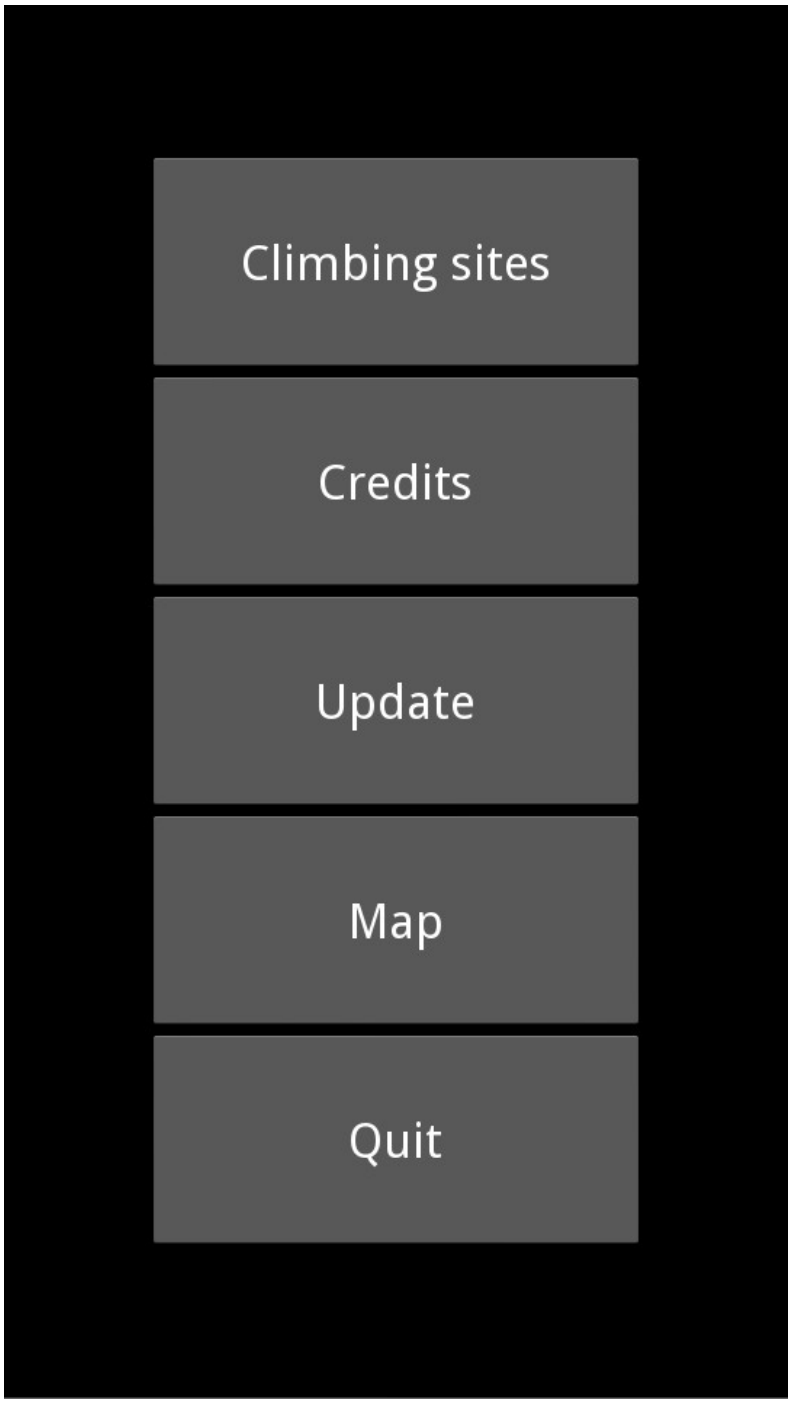
```
<MainScreen>:
    name: 'main'
    size: (360, 480)
    AnchorLayout:
        size_hint: 1, 1
        BoxLayout:
            anchor_x: 'center'
            anchor_y: 'center'
            padding:[10,10]
            spacing: 7
            center_x: self.parent.size[0] /2
            center_y: self.parent.size[1] /2
            size_hint: 0.6, 0.8
            orientation: 'vertical'
            Button:
                pos_hint: {'center_x': 0.5}
                height: root.height*0.15
                width: root.height*0.35
                size_hint:None,None
                text: 'Climbing sites'
                font_size: '30sp'
                on_press: root.manager.transition.direction = 'left'
                on_press: app.root.current = 'database'
                on_press: root.citanje()

            Button:
                pos_hint: {'center_x': 0.5}
                height: root.height*0.15
                width: root.height*0.35
                size_hint:None,None
                #background_normal: 'normal.png'
                #background_down: 'down.png'
                text: 'Settings'
                font_size: '30sp'
                on_press: root.manager.transition.direction = 'left'
                on_release: app.root.current = 'settings'
```

Izvor: autor

Nakon nekoliko stotina linija koda aplikacija je spremna za prve slike. Na slici 4.6 vidljiv je sam dio koda koji obavlja funkciju koja je vidljiva na slici 4.7.

Slika 4.7: Glavni meni aplikacije



Izvor: autor

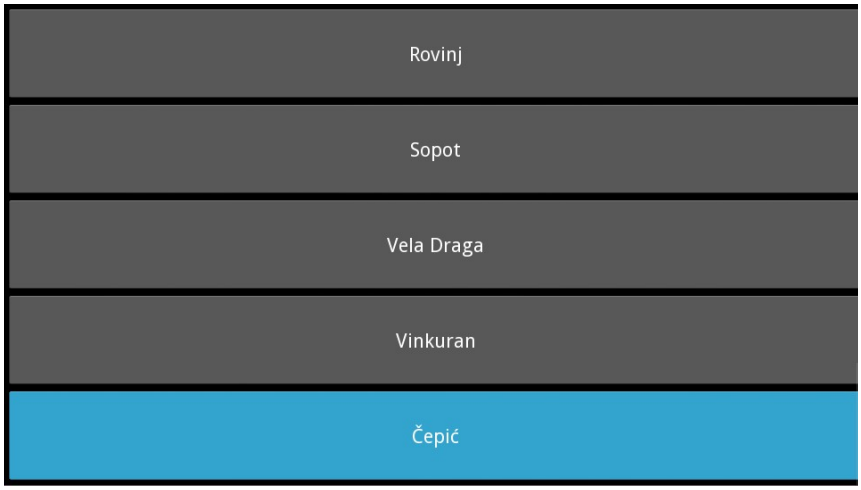
Prvom tipkom ulazimo u popis svih uključenih penjališta u aplikaciji kao što je prikazano na slici 4.8.

Slika 4.8: Popis penjališta.

Izvor: autor

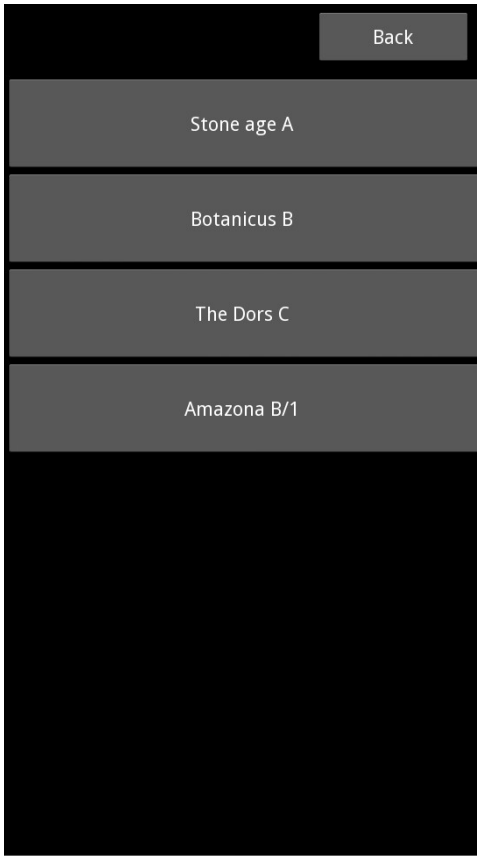
Potrebno je napomenuti kako aplikacija radi također u vertikalnoj i horizontalnoj orijentaciji držanja uređaja (slika 4.9). Sljedeći „podmeni“ apsolutno je neovisan o broju sektora na određenom penjalištu – s obzirom na konstantno otkrivanje novih sektora i čišćenja stijena od raslinja, nedopustivo bi bilo raditi aplikaciju pri svakom otkrivanju novog sektora, stoga je implementirana niska razina automatike koja to radi pri ažuriranju aplikacije s podatcima na serveru (slika 4.10).

Slika 4.9: Horizontalna orijentacija popisa penjališta



Izvor: autor

Slika 4.10: Popis sektora u penjalištu Kamena vrata



Izvor: autor

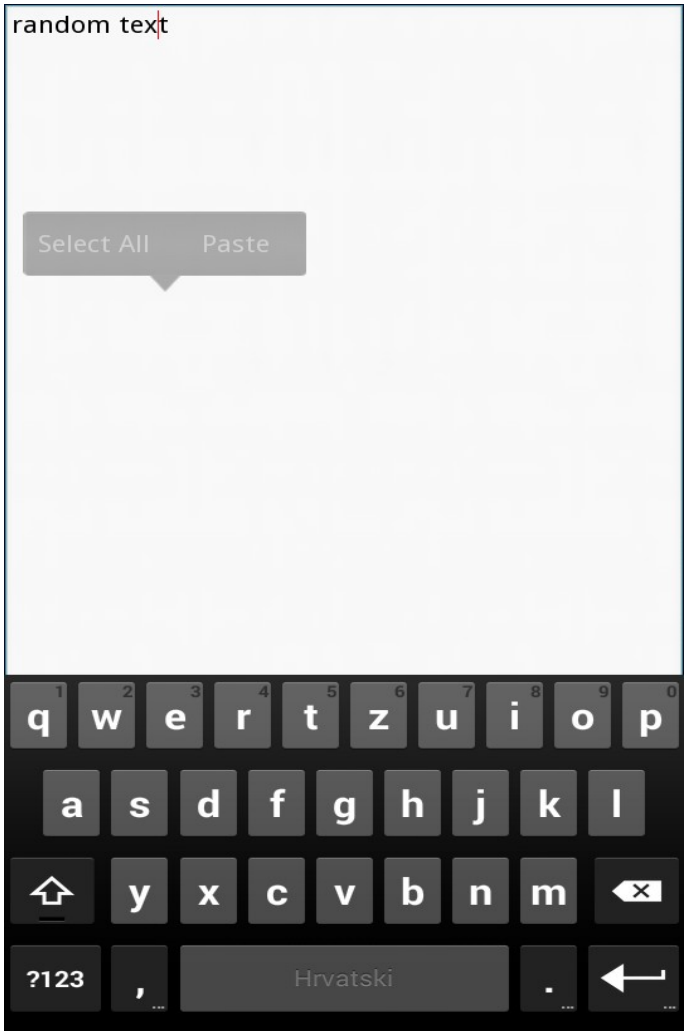
Pritiskom na određeni sektor ulazi se u njega te aplikacija prikazuje smjerove u sektoru (slika 4.11).

Slika 4.11: Popis smjerova u sektoru Botanicus

Izvor: autor

Ulaskom u pojedini smjer aplikacija se pozicionira na ekran na kojem je moguće upisivati tekst (eng. strings), tj. predviđeno je da korisnik upisuje podatke koji su mu relevantni za taj smjer, kao na primjer: količina opreme potrebna za taj smjer, načini savladavanja smjera, smjernice kako prići smjeru itd. Izgled upisa teksta prikazan je na slici 4.12.

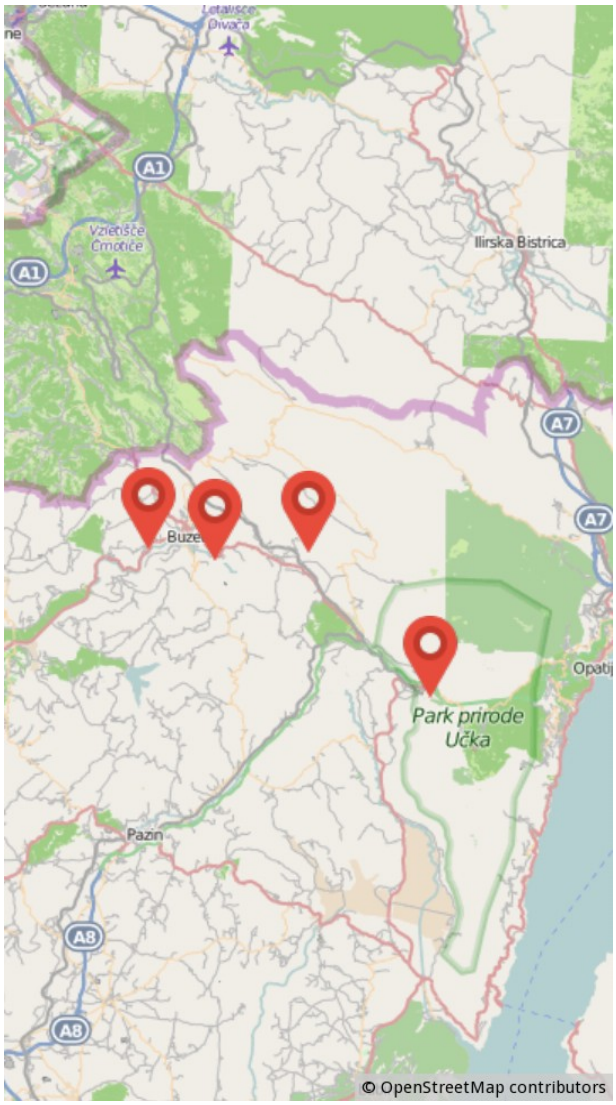
Slika 4.12: Upis teksta relevantnog za smjer



Izvor: autor

Na početnom meniju odabirom na tipku Map, aplikacija nas vodi na dio koji je zadužen za prikaz lokacije penjališta na OpenStreet sloju mape (slika 4.13).

Slika 4.13: Mapa s nekim lokacijama penjališta



Izvor: autor

Odabirom Update opcije na glavnom izborniku aplikacija izvodi trenutno 153 linije koda koje obrađuju xml datoteku na internetskim stranicama plezanje.net te podatke sprema u bazu podataka kako bi se oni izvlačili iz baze i ispisivali na ekran, a izgled Update ekrana prikazan je na slici 4.14.

Slika 4.14: Update ekran

Izvor: autor

5. ZAKLJUČAK

Rapidnim razvojem novih tehnologija, uređaja i softvera, pitanje je vremena kada ćemo se morati prilagođavati drugim tehnologijama i izazovima. Trenutačno je najvažnije biti u toku s postojećom tehnologijom te se služiti njome kako bismo olakšali svakodnevni život i uštedjeli na vremenu, odnosno, kako bismo olakšali radnje što je više moguće s današnjom opremom i tehnologijom.

S obzirom na pokretnost pametnih mobilnih uređaja postoji više načina za izradu aplikacije za pomoć penjačima u pronalasku penjališta, sektora i smjerova. Upotrebom Python programskog jezika, Kivy grafičkog sučelja, a ujedno i jezika, i SQLite baze podataka, moguće je relativno brzo izraditi aplikaciju za pomoć penjačima uz pomoć GIS-a, što potvrđuje postavljenu hipotezu da je uz pravilnu analizu problema moguće kreirati projekt kojim bi se olakšalo snalaženje penjača u penjalištima na području Istre. Treba napomenuti kako podatci u upotrebi moraju biti točni, precizni i ažurni što je više moguće kako bi i aplikacija bila potpuna, točna i ažurna. Baza podataka sadrži 957 smjerova u 19 penjališta, a sadržana je u datoteci od 60 KB na dan 06.07.2015., dok logika programa sadrži 447 linija koda u 16 KB na isti dan, a grafičko sučelje napisano je u 286 linija koda u 8 KB memorije. Bez okvira, odnosno Python interpretera, moguće je sumirati aplikaciju na računalu u manje od jednog MB, što naravno ne znači da će je korisnik moći pokrenuti bez navedenog interpretera, no manje od 1 MB pokreće se na gotovo svim komercijalnim operacijskim sustavima. Za Android operacijski sustav aplikaciju treba kompajlirati zajedno s interpreterom zbog nedostatka Python interpretera u dosadašnjim Android operacijskim sustavima te aplikacija u najboljem slučaju zauzima oko 6 MB, dok ova aplikacija zauzima oko 10 MB.

Ovim završnim radom pokazano je kako je moguće savladati gradivo iz bilo kojeg područja svojevoljno te završiti proces primjenom tog znanja i kako je moguće pronaći besplatne javno dostupne programe, ili čak programe otvorenoga koda, pomoću kojih se može realizirati cilj.

Problemi mogu nastati u održavanju aplikacije ukoliko Android OS u budućnosti ne počne podržavati Python i uključi interpretera u budućim distribucijama, zato što Kivy možda neće biti podržan za Android OS jer cijeli posao odrađuje nekolicina ljudi koja nije plaćena za razvijanje Kivyja, već svojevoljno „vodi borbu s vjetrenjačama“. Što se tiče podataka nema prostora greškama, kao ni odgovornosti za njih jer podatci se preuzimaju s adrese

http://www.plezanje.net/climbing/db/cragIntro.asp?cc=HR&type=C&ord=n i onakvi su kakve korisnik ostavlja na stranici.

Smjer daljnjeg razvoja i proširenja moguć je na ostatak penjališta u Hrvatskoj te čak na ostale susjedne, penjačima privlačne zemlje. Iako iziskuje puno posla, planirano je da aplikacija bude besplatna za korisnike te otvorenog koda kako bi se konstantno unaprijeđivala, ubrzavala, zauzimala manje prostora za pohranu itd.

6. POPIS LITERATURE

Literatura:

1. **GIS – skripte s predavanja**
2. **GIS – bilješke s predavanja**

Internet:

1. **developer.android.com/about/dashboards/index.html**
2. **en.wikipedia.org/wiki/Google**
3. **en.wikipedia.org/wiki/Usage_share_of_operating_systems**
4. **en.wikipedia.org/wiki/Wi-Fi#The_name**
5. **http://developer.android.com/guide/topics/security/permissions.html**
6. **http://developer.android.com/tools/revisions/platforms.html**
7. **https://dspace.mah.se/bitstream/handle/2043/10721/AndroidApplicationDevelopment.pdf?sequence=1**
8. **https://hr.wikipedia.org/wiki/Računalno_programiranje**
9. **http://www.csc.kth.se/utbildning/kandidatexjobb/teknikmanagement/2010/rapport/grundstrom_peter_K10054.pdf**
10. **http://kivy.org/planet/**
11. **https://www.python.org/**

7. POPIS SLIKA I GRAFIKONA

Slika 2.1: Simon Personal Communicator....9
Slika 2.2: Model Nokia 9000....10
Slika 2.3: Model Kyocera 6035....11
Slika 2.4: Graffiti writing....12
Slika 2.5: Google G1....14
Slika 2.6: Shematski prikaz arhitekture Android platforme....15
Slika 2.7: Grafički prikaz dostupnosti pojedinih API-ja kroz vrijeme....17
Slika 3.1: Razlika između rasterske i vektorske slike....22
Slika 4.1: Dijagram toka za glavni izbornik....33
Slika 4.2: Dijagram toka za ekran Climbing sites....34
Slika 4.3: Python kod u aplikaciji primjenom GIS-a u penjanju....37
Slika 4.4: SQLite baza podataka za izradu aplikacije primjenom GIS-a u penjanju....39
Slika 4.5: Kivyjem podržani sustavi....40
Slika 4.6: Prikaz koda u Kivy jeziku....41
Slika 4.7: Glavni meni aplikacije....42
Slika 4.8: Popis penjališta....43
Slika 4.9: Horizontalna orijentacija popisa penjališta....44
Slika 4.10: Popis sektora u penjalištu Kamena vrata....44
Slika 4.11: Popis smjerova u sektoru Botanicus....45
Slika 4.12: Upis teksta relevantnog za smjer....46
Slika 4.13: Mapa s nekim lokacijama penjališta....47
Slika 4.14: Update ekran....48