

**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
Одсек за рачунарство и аутоматику
Катедра за рачунарску технику и рачунарске комуникације**

ДИПЛОМСКИ - БЕЧЕЛОВ РАД

Кандидат: Душан Живков
Број индекса: Е11021

Тема рада: Један приступ развоју сервиса у
мобилној телефонији користећи „Parlay“ спрегу

Ментор рада: Др Никола Теслић

Нови Сад, јун 2008.

	UNIVERZITET U NOVOM SADU • FAKULTET TEHNIČKIH NAUKA 21000 NOVI SAD, Trg Dositeja Obradovića 6
	KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj, RBR:			
Identifikacioni broj, IBR:			
Tip dokumentacije, TD:		Monografska publikacija	
Tip zapisa, TZ:		Tekstualni štampani materijal	
Vrsta rada, VR:		Diplomski-ba rad	
Autor, AU:		Dušan Živkov	
Mentor, MN:		Dr Nikola Teslić	
Naslov rada, NR:		Jedan pristup razvoju servisa u mobilnoj telefoniji koristeći Parlay spregu	
Jezik publikacije, JP:		Srpski	

Jezik izvoda, JI :		Srpski	
Zemlja publikovanja, ZP :		Srbija	
Uže geografsko područje, UGP :		Vojvodina	
Godina, GO :		2008	
Izdavač, IZ :			
Mesto i adresa, MA :			
Fizički opis rada, FO : (poglavlja/strana/citata/tabela/slika/grafika/priloga)			
Naučna oblast, NO :		Elektrotehnika i računarska tehnika	
Naučna disciplina, ND :			
Predmetna odrednica/Ključne reči, PO :			
UDK			
Čuva se, ČU :		U biblioteci FTN, Novi Sad, Trg Dositeja Obradovića 6	
Važna napomena, VN :			

Izvod, IZ:			
Datum prihvatanja teme, DP:			
Datum odbrane, DO:			
Članovi komisije, KO:	Predsednik:		
	Član:		Potpis mentora
	Član, mentor:		

	UNIVERSITY OF NOVI SAD • FACULTY TECHNICAL SCIENCES 21000 NOVI SAD, Trg Dositeja Obradovića 6
	KEY WORDS DOCUMENTATION

Accession number, ANO:			
Identification number, INO:			
Document type, DT:		Monographic's publication	

Type of record, TR:		Word printed record	
Contents code, CC:		Diploma-Bechlor thesis	
Author, AU:		Dušan Živkov	
Mentor, MN:		Ph. D.E.E. Nikola Teslić	
Title, TI:		One solution for developing services in mobile telephony usingi Parlay interface	
Language of text, LT:		Serbian	
Language of abstract, LA:		Serbian	
Country of publication, CP:		Serbia	
Locality of publication, LP:		Vojvodina	
Publication year, PY:		2008	
Publisher, PB:			
Publication place, PP:			

Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendixes)			
Scientific field, SF:		Electrical engineering and computer science	
Scientific discipline, SD:			
Subject/Key words, S/KW:			
UC			
Holding data, HD:		Library of FTN, Novi Sad, Trg Dositeja Obradovića 6	
Note, N:			
Abstract, AB:			
Accepted by the Scientific Board on, ASB:			
Defended on, DE:			
Defended Board, DB:	President:		

	Member:		Mentor's sign
	Member, Mentor:		

SKRAĆENICE:

NRG – Network Resources Gateway

SMS – Short Messages Service

MMS – Multimedia Messaging Service

WAP – Wireless Application Protocol

SDK – Software Development Kit

API - Application programming interface

OSA – Open Service Access

H-OSA – High-Level OSA

SQL – Structured Query Language

SCS – Service Capability Server

PSTN – Public Switched Telephone Network

GPRS – General Packet Radio Service

GSM – Global System for Mobile

WCDMA – Wideband Code Division Multiple Access

CORBA – Common Object Request Broker Architecture

ORB – Object Request Broker

IOP – Internet Inter-ORB Protocol

IDL – Interface Description Language

ATT – Automatic Testing Tool

1. Sadržaj

2. 1. UVOD

Zadatak je iz oblasti mobilne telefonije, i njegov cilje je upoznavanje sa razvojom usluga koje se oslanjaju na Parlay/OSA spregu koristeći mogućnosti koje pruža Eriksonov NRG (Ericsson Network Recourse Gateway). Pored upoznavanja cilj je i uspešan razvoj programa koji će demonstrirati neke od mogućnosti koje pruža NRG i njihovu praktičnu primenu. U razvoju će se koristiti Eriksonov NRG vezija 4.1 IDM 10. Kao primer je izabran program koji treba da omogući obaveštavanje korisnika putem SMS-a o lokacijama koje se nalaze u njihovoj neposrednoj blizini te o uslugama koje te lokacije pružaju, a od interesa su za korisnika. Korisnik

ima mogućnost izbora usluga koje ga zanimaju. Program treba razviti u programskom jeziku Java. Razlog zašto je izabran programski jezik Java je taj što, uz NRG, Erikson je razvio i NRG SDK koji u sebi sadrži Java biblioteke, simulator, automatski alat za testiranje i dokumentaciju.

3. **2. TEORIJSKE OSNOVE**

U ovom poglavlju biće opisani programski alati i standardi koji su korišćeni u izradi.

1. **2.1. OSA koncept**

Osnova celog zadatka se oslanja na OSA (Open Service Access) koncept. OSA koncept se sastoji od tri sloja. Prvi nivo je nivo usluga unutar kojega je smešten NRG kao mrežni prolaz za programe. U kontrolnom nivo i nivou veze su smešteni elementi telekomunikacione mreže. U kontrolnom nivou se odvija signalizacija, a u nivou veze se odvija razmena podataka. Prednost ovakve strukture je u tome što NRG znatno pojednostavljuje pristup resursima telekomunikacione mreže, sakrivajući složenost mreže, čime se i smanjuje potrebno znanje da bi se mogli koristiti resursi mreže.



Sl. 2.1 OSA koncept

2. **2.2. OSA/Parlay programska sprega**

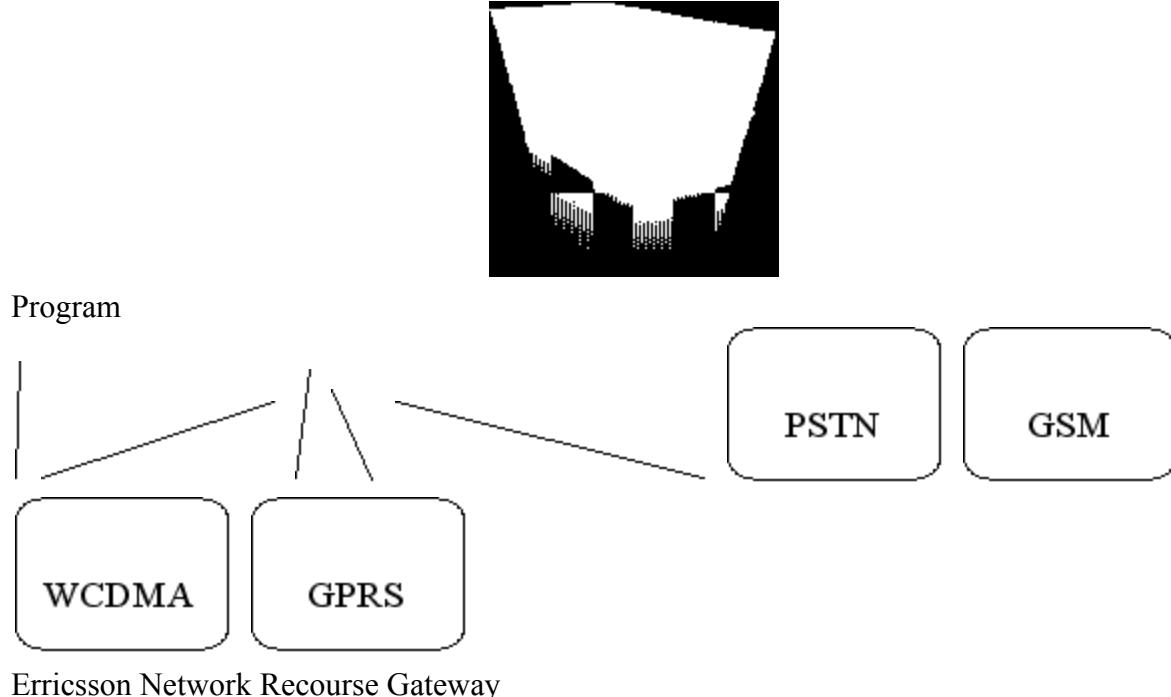
Ova programska sprega se koristi da bi se program povezao sa nivoom usluga, tj. sa NRG-om.

Sva komunikacija između programa i NRG-a se odvija preko ove sprege. Neki od primera koje se usluge sve mogu razviti korišćenjem ove sprege su:

- Dobijanje pristupa uslugama zahtevajući ih od Framework-a
- Stvaranje i preusmeravanje poziva, kao i puštanje obaveštenja i preuzimanje unetih brojeva od strane korisnika
- Zahtevanje pozicije i statusa mobilnih telefona
- Slanje i prijem svih vrsta poruka (SMS, MMS, Wap Push)

3. 2.3. Ericsson Network Resource Gateway

Glavni deo NRG-a jeste Service Capability Server (SCS). SCS omogućava programu da koristi mrežne usluge na siguran način. Iza SCS-a se nalazi telekomunikaciona mreža koja koristi široku lepezu protokola da bi pružala usluge (PSTN, GSM, GPRS, WCDMA).



Sl. 2.2 Princip rada NRG-a

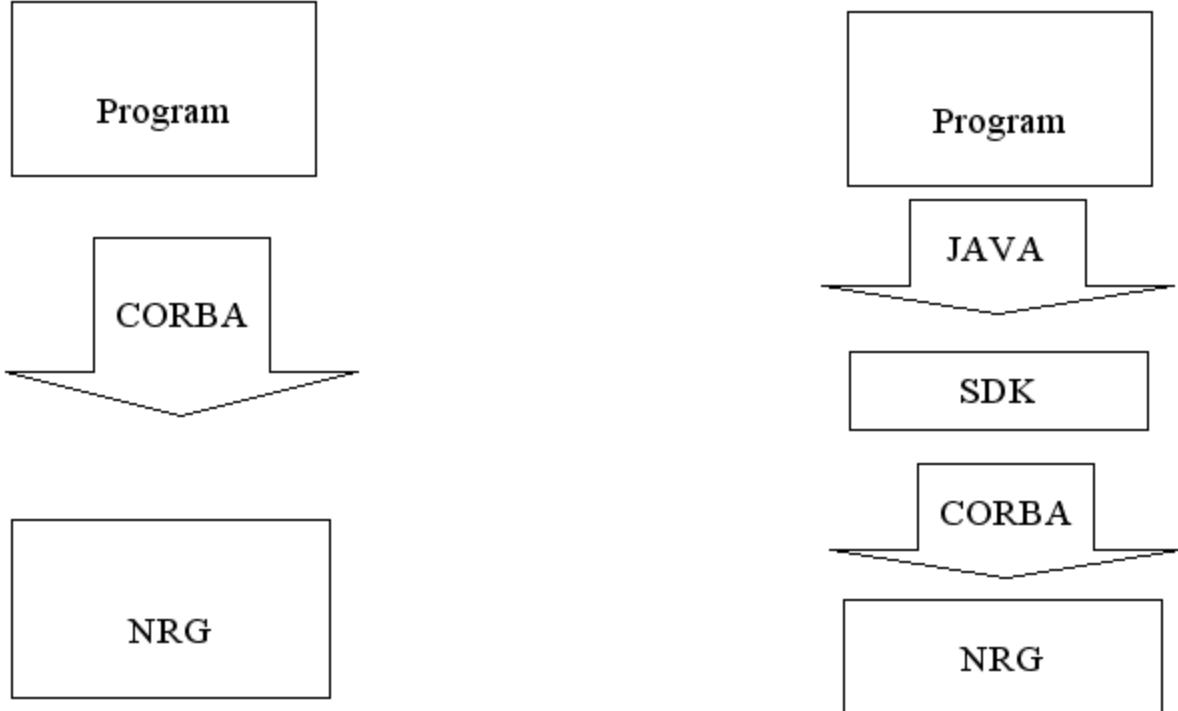
4. 2.4. NRG SDK

SDK nudi biblioteke za razvoj programa u JAVA programskom jeziku. Kako NRG omogućava jednostavniji pristup resursima mreže sakrivajući pozadinske protokole, tako i SDK dodatno pojednostavljuje upotrebu sakrivajući složenu CORBA komunikaciju. SDK se sastoji iz tri dela:

- Core API
- Test API
- Utility API

U sklopu SDK se još nalaze simulator i alat za automatsko testiranje. Alat za automatsko testiranje simulira NRG i automatski generiše mrežne aktivnosti. Za razliku od njega simulator poseduje grafičku spregu koji omogućava koristi pristupačniju kontrolu događaja na mreži.

Sledeća slika ilustruje pristup NRG sa i bez SDK.



Sl 2.3a Pristup bez SDK Sl 2.3b Pristup pomoću SDK

5. **2.5. JAVA**

Tokom izrade je korišćena JAVA 1.3.1_15, jer je to poslednja verzija na kojoj je testiran NRG SDK, , i samo na njoj funkcioniše simulator i klase koje dolaze u sklopu SDK .

6. **2.6. CORBA (Common Object Request Broker Architecture)**

Komunikacija programa sa NRG-om se odvija korišćenjem ORB-a(Object Request Broker) preko IIOP protokola (Internet Inter-ORB Protocol). ORB koji se koristi mora biti kompatibilan sa CORBA 2.3. Osnovna odlika CORBA jeste što se koristeći IIOP omogućava komunikacija programa razvijenih u raznim programskim jezicima, i na različitim operativnim sistemima.

7. **2.7. MySQL**

MySQL je višenitni sistem za rukovanje bazama podataka sa podrškom za više korisnika. Korišćenje baza podataka i poslužioca MySQL nije definisano zadatkom, ali je znatno olakšalo pristup i kontrolu podataka.

4. **3. ANALIZA PROBLEMA**

1. **3.1. CORBA**

Ključna osnova ovog programa je CORBA, i razumevanje njenih principa je važno. CORBA je mehanizam koji se koristi za pozivanje metoda između programskih objekata. CORBA koristi IDL (Interface Definition Language) da bi definisao spregu preko koje će se objekti predstaviti spoljnom svetu. CORBA takođe specificira mapiranje iz IDL jezika u neki jezik koji koristi program (C++, Java, Ada...). Slika demonstrira taj princip:

Sl 3.1. Ilustracija CORBA

Na konkretnom zadatku, iako se može koristiti direktni pristup NRG-u pomoću CORBA, koristi se jednostavniji način koristeći NRG SDK u sklopu kojeg se nalazi framework koji sakriva CORBA komunikaciju svojim metodama. Jedna od ključnih stvari koje CORBA omogućava jeste pozivanje metoda objekata koji nisu inicijalizovani lokalno (u programu) i njihovo izvršavanje se obavlja na mestu gde je objekat inicijalizovan. Na ovaj način program komunicira sa NRG-om. Program poziva metode u NRG-u, koristeći upravljače servisa (service managers) i framework, i tako šalje zahteve NRG-u, a NRG odgovara programu pozivajući takozvane callback metode prethodno naznačenih objekata.

2. **3.2. Simulator**

Ograničenost simulatora je bila velika prepreka prilikom izrade. Pošto se veliki deo zadatka oslanja na usluge za lociranje mobilnih telefona, problem se javio jer su mogućnosti simulatora ograničene kada su u pitanju usluge ovog tipa. Program bi trebao da vrši praćenje pozicije svih

pretplaćenih korisnika i izračunava rastojanje od interesnih tačaka. Prema Parlay/OSA specifikacijama predviđeno je slanje zahteva za obaveštavanja programa u slučaju ulaska korisnika unutar određene zone oko neke tačke. Nažalost ova opcija nije realizovana u sklopu simulatora. Takođe prema H-OSA programskoj sprezi (nadskup OSA programske sprege, dizajniran od strane Eriksona) trebalo bi da postoji mogućnost zahtevanja periodičnog dobijanja pozicije korisnika od strane NRG-a, ali ni ovo nije realizovano u simulatoru. Ovim ograničenjima je na program prenesena dodatna obrada čime je povećana kompleksnost programa. Problem određivanja da li je korisnik u blizini neke interesne tačke je rešen tako što je ta dodatna obrada prenesena na poslužioc MySQL.

Sl 3.2. Izgled simulatora

3.3. Baze podataka

Prethodno naveden problem je i jedan od razloga zašto je čuvanje podataka prenesno u baze podataka tj. na poslužioc MySQL. Drugi razlog jeste sigurniji pristup i upis podataka kao i brže dobijanje željenih informacija. Naravno sa uvođenjem baza podataka, javlja se i problem pravilnog dizajniranja istih, da bi se mogle iskoristiti sve prednosti koje baze podataka pružaju. Analizom podataka dobijen je model baze unutar koje se nalaze sledeće tabele:

- users – tabela u koji su smešteni podaci o korisnicima
- services – tabela u koju su smešteni podaci o lokacijama (servisima)
- service info – tabela koja povezuje korisnika sa određenim ključnim rečima koje ga interesuju
- services info – tabela koja povezuje lokacije sa ključnim rečima koje one pokrivaju
- user service info – tabela koja sadrži informacije o posetama korisnika na lokacije

3.3 Izgled baze

3.3.1. Tabela USERS

U ovu tabelu su smešteni svi korisnici koji se bar jednom aktiviraju.

Unutar tabele Users se nalaze sledeća polja:

- User_ID – jedinstvena identifikacija svakog korisnika
- Phone_number – telefonski broj korisnika
- Name – ime korisnika
- User_longitude – longituda poslednje korisnikove pozicije
- User_latitude – latituda poslednje korisnikove pozicije
- Active – indikator da li je korisnik aktivan

3.3.2 .Tabela SERVICES

U ovu tabelu su smeštene sve lokacije koje pružaju usluge.

Unutar tabele Services se nalaze sledeća polja:

- Service_ID – jedinstvena identifikacija lokacije
- Name – ime lokacije
- Service_longitude – longituda pozicije lokacije
- Service_latitude – latituda pozicije lokacije

3.3.3. Tabela SERVICE INFO

U ovu tabelu su smešteni svi parovi lokacija-usluga koju pruža ta lokacija (jedna lokacija može

da pruža više usluga).

Unutar tabele Services se nalaze sledeća polja:

- Service_info_ID – jedinstvena identifikacija
- Service_ID – identifikacija lokacije koja pruža uslugu
- Keyword – naziv usluge koju lokacija pruža
- Description – poruka koja će biti prosleđena korisniku koji se nađe u blizini lokacije, a zainteresovan je za uslugu

3.3.4. Tabela USER INFO

U ovu tabelu su uneseni parovi korisnik – usluga koji nam govore na koje usluge je korisnik pretplaćen.

Unutar tabele Services se nalaze sledeća polja:

- User_info_ID – jedinstvena identifikacija
- User_ID – identifikacija korisnika
- keyword – naziv usluge na koju se korisnik pretplatio

3.3.5. Tabela USER SERVICE INFO

U ovu tabelu su smešteni parovi lokacija – korisnik i informacija o poslednjoj poseti korisnika lokaciji.

Unutar tabele Services se nalaze sledeća polja:

- User_Service_info_ID – jedinstvena identifikacija
- User_ID – identifikacija korisnika
- Service_ID – identifikacija lokacije
- Last_visit – vreme poslednje posete korisnika lokaciji

5. 4. REŠENJE PROBLEMA

U ovom poglavlju će biti opisani programski moduli koji su realizovani u rešenju, njihova međusobna interakcija i način korišćenja programa.

1. 4.1. Konfiguraciona datoteka

Ovde su opisani ulazni parametri koji se nalaze u konfiguracionoj datoteci *locationdetect.ini*:

- *CosNamingResponder* – Adresa NRG-a
 - host – IP adresa NRG-a
 - port – port na kojem NRG osluškuje
- *app.id* – identifikator aplikacije (trenutno nije moguće ga promeniti zato što simulator dozvoljava samo programima sa imenom *test_application* da se spoje na NRG)
- *map* – informacije o mapi
 - *longitude1* – longituda gornje leve tačke mape
 - *latitude1* – latituda gornje leve tačke mape
 - *longitude2* – longituda donje desne tačke mape
 - *latitude2* – latituda donje desne tačke mape
- *url* – adresa poslužioca MySQL
- *user* – ime korisnika na poslužiocu MySQL
- *password* – šifra korisnika na poslužiocu MySQL
- *DBname* – ime baze u koju su smešteni podaci

- *service_number* – broj na koji se korisnici pretplaćuju za informacije o ključnim rečima
- *distance* – rastojanje na kojem korisnik mora biti da bi dobio informaciju o lokaciji (u stepenima)
- *start_phone* – prvi broj grupe korisnika koji su pretplaćeni na uslugu
- *user_count* – broj korisnika koji su pretplaćeni na uslugu
- *interval* – veličina intervala u kom će se vršiti prozivka pozicija aktivnih korisnika
- *unit* – jedinica u kojoj je unesen interval. Može biti : SECOND,MINUTE,HOUR,DAY,WEEK,MONTH,YEAR
- *service* – informacije o lokacijama koje se smeštaju u bazu
 - *name* – ime lokacije
 - *longitude* – longituda lokacije
 - *latitude* – latituda lokacije
 - *info* – informacije o uslugama koje pruža lokacija
 - *keyword* – ključna reč za korisnika (**null** ako je obavezno obaveštavanje)
 - *description* – opis usluge koji će biti poslat korisniku

2. 4.2. Opis koda

Program se sastoji od sledećih klasa:

- LocationDetect
- Configuration
- UserStatusTracker
- LocationTracker
- Messenger
- LocatingThread
- Map

Sledeća slika predstavlja UML dijagram klasa:

Sl 4.1. UML dijagram

4.2.1. Klasa LocationDetect

Ova klasa je glavna klasa. Ona sadrži metodu *main* sa kojom kreće izvršavanje programa.U *main* metodi se poziva konstruktor ove klase u čijem telu se pokreće program. Prvo se poziva metoda *load* klase *Configuration* koja obavlja učitavanje ulaznih parametara. Nakon ovoga sledi povezivanje na poslužioc MySQL, iscrtavanje GUI, pokretanje metode za popunjavanje baze lokacijama. Nakon ovoga instanciraju se objekti ostalih klasa, program se povezuje na NRG (preko *FWProxy* klase), započinje obaveštavanje o promenama statusa korisnika i obaveštavanje o pristiglim porukama na servisni broj. Na kraju se pokreće nit koja periodično zahteva pozicije korisnika, kao i nit koja osvežava mapu.

Klasa *LocationDetect* poseduje i jednu metodu *dispose* koje sa poziva pre izlaska iz programa da bi se oslobođili svi zauzeti resursi.

1. 4.2.2. Klasa Configuration

Zadatak ove klase jeste učitavanje i pamćenje ulaznih parametara kako bi se mogli proslediti dalje u programu. Klasa nasleđuje *NestedProperties* klasu koja je deo NRG SDK i ima metode za učitavanje parametara iz podešene konfiguracione datoteke. Pored ovoga ova klasa poseduje i metodu *PopulateDB* koja popunjava bazu podataka sa lokacijama iz konfiguracione datoteke. U

slučaju da je neki od parametara pogrešno formulisan, ova klasa baca izuzetak koji detektuje glavna klasa i obustavlja se rad programa.

2. 4.2.3. Klasa UserStatusTracker

Ova klasa je zadužena za praćenje statusa korisnika. Ovo je potrebno da bi se optimizovala obrada, jer izbegavamo proveru pozicije za neaktivne korisnike. Unutar ove klase se se srećemo prvi put sa upravljačem servisa, u konkretnom slučaju, to je za usluge koje su u vezi sa statusom korisnika. Ovaj upravljač zapravo predstavlja izlaznu tačku preko koje mi komuniciramo sa NRG-om i zahtevamo usluge. Zahtevanje usluga se u ovoj klasi obavlja u telu metode *startTriggering*, kojoj se prosleđuje početni broj grupe korisnika i veličinu grupe korisnika. Na kraju metode, kada su svi podaci spremni, poziva se metoda *triggeredStatusReportingStartReq* koja javlja NRG-u da počne da obaveštava program o svim promenama statusa za zatražene korisnike. NRG program obaveštava tako što poziva *triggeredStatusReport* metodu ove klase, jer je ova klasa prosleđena kao callback klasa. Da bi ova klasa mogla da bude callback klasa mora da realizuje sučelje *IpAppUserStatus* što i radi. Ova klasa takođe nasleđuje *IpAppUserStatusAdapter* da bi se izbegla potreba za definisanjem svih metoda sučelja *IpAppUserStatus*. Kada dođe do promene statusa proverava se da li je korisnik postao aktivan. U slučaju da jeste, sledi provera da li je već u bazi, ako nije ubacuje se, ako jeste samo mu se menja status. Kada korisnik postane nedostupan proverava se da li je u bazi, i ,ako jeste, menja mu se status.

3. 4.2.4. Klasa LocationTracker

Ova klase je najvažniji deo programa jer se u sklopu nje obavljaju najsloženiji upiti za bazu podataka, kako bi se dobio odgovor da li je korisnik blizu lokacije od interesa i da bi se dobili svi potrebni podaci da se on obavesti. Ova klasa nasleđuje *IpAppUserLocationAdapter* i realizuje sučelje *IpAppUserLocation* iz istih razloga kao i prethodna. Metoda *getLocation* se periodično poziva kako bi se od NRG-a dobili podaci o pozicijama korisnika. NRG izveštaj sa pozicijama korisnika prosleđuje programu kao jedan od parametara prilikom poziva metode *extendedLocationReportRes* gde se za svaki validan izveštaj o poziciji poziva metoda *notifyLocation*. Ovde postoji i sinhoronizujući objekat *LOCK* koji sprečava početak obrade novog izveštaja sve dok se obrada prethodnog ne završi. Metoda *notifyLocation* obavlja najveći deo obrade. Prvo što se obavi u ovoj metodi jeste osvežavanje koordinata korisnika u bazi podataka. Nakon toga sledi niz upita kojima se dobija korisnička identifikacija u bazi i sve lokacije koje pružaju usluge od interesa za korisnika, a nalaze se u zadatoj blizini. Kada dobijemo ove podatke, ispitujemo u bazi da li je korisnik već posetio ovu lokaciju i ako jeste, da li je prošao zadati vremenski interval od poslednje posete. Kada dobijemo odgovore, zaključujemo da li je potrebno obavestiti korisnika o nekoj usluzi i postavaljamo upit bazi kojim dobijamo informacije koje ćemo proslediti korisniku. Ove informacije šaljemo korisniku putem SMS-a pomoću klase *Messenger*, tačnije njene metode *sendSMS* i na mapi palimo signalni znak da je poruka poslata. Kada je korisnik obavešten, u bazi se osvežavaju informacije o poslednjoj poseti tj. unose se ako nisu postojale.

4. 4.2.5. Klasa Messenger

Zadatak ove klase je kontrola SMS saobraćaja. Preciznije rečeno kada se otkrije da se korisnik našao u neposrednoj blizini neke lokacije koja pruža usluge od interesa za korisnika, preko ove klase se vrši slanje kratkih poruka korisnika. Metoda koja se poziva da bi se poslala poruka je *sendSMS* . Parametri ove metode su broj sa koga se šalje poruka(broj servisa), broj na koji se šalje poruka (broj korisnika) i sadržaj poruke (zavisno od usluge i lokacije). Ova metoda je

prlično jednostavna i svodi se na pripremu parametara i pozivanje metode *hosaSendMessageReq* upravljača usluge za interakciju sa korisnicima (on se koristi za slanje poruka). Kao odgovor NRG poziva metodu *hosaSendMessageRes* preko koje prosleđuju informacije da li je slanje obavljeno uspešno. Da bi se NRG-u omogućilo pozivanje ove metode klasa *Messenger* nasleđuje *IpAppHosaUIManagerAdapter* realizuje sučelje *IpAppHosaUIManager*. Pored slanja poruka ova klasa ima još jedan važan zadatak, a to je prijem i obrada poruka koje su poslate na broj servisa. Za početak se pošalje zahtev NRG-u da se započne prosleđivanje poruka, poslatih na broj servisa, ka programu. Ovo se obavlja u metodi *startNotifications*. Nakon ovoga NRG obaveštava program o pristiglim porukama pozivanjem metode *reportNotification*. Kada se primi poruka ona se parsira da bi se utvrdilo da li je ispravno formulisana i na osnovu sadržaja se obavi zahtev. Posotje dve mogućnosti: prijava na informacije o usluzi i odjava na informacije o usluzi. Izgled poruke za prijavu je: **REG naziv_usluge**, a izgled poruke za odjavu je: **UNREG naziv_usluge**. Korisnik se na kraju obaveštava da li se uspešno prijavio tj. odjavio (ili nema tu mogućnost ili je pogrešno formulisao poruku).

5. **4.2.6. Klasa LocatingThread**

Zadatak ove klase jeste da periodično zahteva izveštaje o poziciji korisnika. Ovo se sve obavlja u *run* metodi ove klase. Prvo se od baze zahteva spisak svih aktivnih korisnika i zatim se poziva metoda *LocationTracker.getLocation* za dobijene korisnike. Druga metoda koja postoji u ovoj klasi jeste *stopLocating* koja se poziva na izlasku iz programa, sa ciljem da se zaustavi ova nit. U klasi postoji i sinhronizujući objekat *LOCK* koji služi da bi se sprečio istovremeni pristup promenjivoj *terminate* koja služi za signalizaciju kraja rada. Klasa nasleđuje klasu *Thread* da bi mogla da se pokrene kao zasebna nit. Program tj. ova klasa svake 3 sekunde zahteva izveštaje o pozicijama korisnika koji, kada pristignu, obrađuje klasa *LocationTracker*.

6. **4.2.7. Klasa Map**

Jedini zadatak ove klase jeste da korisniku dočara približni prikaz događanja u realnom svetu, tačnije rečeno da iscrta pozicije korisnika i lokacija, i grafički dočara slanje SMS poruka korisnicima. Ova klasa realizuje sučelje *Runnable* zato što se jedan njen deo pokreće kao zasebna nit. U sklopu konstruktora klase se nalazi podešavanje izgleda mape i učitavanje svih lokacija iz baze, a zatim i njihovo iscrtavanje pomoću metode *drawService*. Ovoj metodi se prosleđuje ime lokacije i njene koordinate, a ona ga dodaje na odgovarajuće mesto na mapi. Deo koji radi kao nit se nalazi u sklopu metode *run*. Ovaj deo se odnosi na periodično pozivanje iz baze informacija o aktivnim korisnicima i njihovo iscrtavanje na mapi koristeći metodu *drawUser* koja radi po sličnom principu kao i *drawService* samo što se iscrtava drugaćija ikonica. Pored ovih metoda, klasa *Map* poseduje i metodu *blinkMessage* koja se poziva kad god se korisniku pošalje poruka o informaciji za neku uslugu. Ova metoda iscrtava malu ikonicu poruke na poziciji gde je korisnik primio poruku. U klasi postoje još metode *toPixels* koje služe za konverziju iz koordinata mape (stepeni longitude i latitude) u koordinate slike (pikseli).



Sl. 4.2. Izgled mape

6. **TESTIRANJE**

Testiranje programa je jedna od najvažnijih stavki u njegovom razvoju. Tokom testiranja korišćena su dva pristupa. U prvom pristupu korišćen je simulator i proveravani su određeni scenariji dogadaja, te da li se odvijaju kako je pretpostavljeno. Drugi pristup jeste puštanje programa da radi sa alatom za automatsko testiranje (ATT) i provera za neočekivanja ponašanja. Za sve testove korišćena ista konfiguraciona datoteka (osim za test 10) i ona izgleda ovako:

```
CosNamingResponder.0.host=localhost  
CosNamingResponder.0.port=23104  
CosNamingResponder.1.host=localhost  
CosNamingResponder.1.port=23104
```

```
app.id=test_application1
```

```
map.longitude1 = 0  
map.latitude1 = 0  
map.longitude2 = 1  
map.latitude2 = 1
```

```
url = jdbc:mysql://localhost:3306/  
user = Parlay  
password = Ericsson  
DBname= locationdetector
```

service_number = 8888
distance = 0.1

start_phone = 6612300
user_count = 10

interval = 3
unit = MINUTE

service.0.name = Kantina Krisina
service.0.longitude = 0.18
service.0.latitude = 0.595
service.0.info.0.keyword = pivo
service.0.info.0.description = Svatite kod nas na pivo
service.0.info.1.keyword = sendvic
service.0.info.1.description = Svatite kod nas na sendvic

1. service.2.name = Orfej
service.2.longitude = 0.88
service.2.latitude = 0.3
service.2.info.0.keyword = kafa
service.2.info.0.description = Svatite kod nas na kafa

service.3.name = 10
service.3.longitude = 0.02
service.3.latitude = 0.56
service.3.info.0.keyword = sok
service.3.info.0.description = Svatite kod nas na sok
service.3.info.1.keyword = rucak
service.3.info.1.description = Svatite kod nas na rucak

service.4.name = Sala
service.4.longitude = 0.35
service.4.latitude = 0.88
service.4.info.0.keyword = sport
service.4.info.0.description = Svatite kod nas na terene

service.5.name = Ekonomski faks

```
service.5.longitude = 0.68  
service.5.latitude = 0.75  
service.5.info.0.keyword = null  
service.5.info.0.description = Svatite kod nas Obavezno  
2.  
3.  
4.  
5.  
6.  
7.  
8.  
9.  
10.  
11.
```

12. **5.1. Testiranje sa simulatorom**

Ovde su opisani scenariji testova sa simulatorom.

1. **5.1.1. Test 1**

Opis:

U ovom testu na mrežu će se aktivirati novi korisnik. Cilj testa jeste provera da mapa uspešno prikazuje nove korisnike i da se oni registruju u bazi.

Preduslov:

Pokrenut poslužioc MySQL i pokrenut simulator NRG-a. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Pokretanje programskog rešenja.

Sl. 5.1 Prazna mapa

2. Aktiviranje novog korisnika u simulatoru NRG-a.

Sl. 5.2. Dodavanje novog korisnika

Očekivano ponašanje:

Ikonica koja predstavlja korisnika se pojavlje na mapi u programu. U bazi podataka se upisuju podaci o korisniku.



Sl 5.3. Mapa sa detektovanim novim korisnikom

User_ID	Phone_number	Name	User_longitude	User_latitude	Active
1	6612300	HULL	0.5450000166...	0.1074999943...	1

Sl 5.4. Izgled tabele korisnika nakon aktiviranja novog korisnika

Test prošao:

DA.

2. 5.1.2. Test 2

Opis:

U ovom testu registrovani korisnik prolazi pored mesta na kojem je obavezno obaveštavanje svih korisnika. Cilj testa je da proveri ispravnost sistema za obaveštavanje korisnika i grafičkog prikaza slanja kratkih poruka. Kada korisnik prođe pored obavezne lokacije dobija poruku : „Micronas: Svatite kod nas OBAVEZNO!“.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. . Baza podataka mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje jednog korisnika u simulatoru NRG-a
2. Korisnika počinje da se kreće ka poziciji na kojoj se nalazi lokacija sa obaveznim obaveštavanjem.

Sl. 5.5. Obavezna lokacija

Očekivano ponašanje:

Korisnik prima SMS poruku od programa i otvara je na svom mobilnom uređaju. Sadržaj poruke je opis usluga koje pruža lokacija.



Sl. 5.6. Pristigla poruka

Test prošao:

DA

3. 5.1.3. Test 3

Opis:

Ovaj test proverava deo programa koji je zadužen za registrovanje korisnika na određenu uslugu. Korisnik šalje poruku sa tekstom: „REG sok“ na broj 8888 (to je broj servisa) i kao odgovor se dobija: „Uspesno ste se registrovali na informacije o : sok“ i u bazi se to beleži (slike u testu 4 ilustruju i ovaj test).

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. Baza podataka mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje jednog korisnika u simulatoru NRG-a
2. Korisnik mobilnog uređaja pokreće aplikaciju za slanje kratkih poruka i na broj 8888 šalje poruku sa tekstrom "REG sok".

Očekivano ponašanje:

Korisnik prima SMS poruku od programa i otvara je na svom mobilnom uređaju. Sadržaj poruke je „Uspesno ste se registrovali na informacije o : sok“. U bazi se beleži korisnikova registracija na željenu ključnu reč.

Test prošao:

DA.

4.

5.

6. 5.1.4. Test 4

Opis:

Ovaj test se je vrlo sličan prethodnom, ali se ovde korisnik registruje na uslugu koju trenutno ni jedna lokacija ne pruža. Prvo se šalje poruka : „REG vecera“ na 8888, a program odgovara korisniku sa: „Registrovali ste se na: vecera. Ali нико trenutno ne pruza uslugu“. Unos se takođe beleži u bazi.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenja zadatka.

Koraci:

1. Akiviranje jednog korisnika u simulatoru NRG-a
2. Korisnik mobilnog uređaja pokreće aplikaciju za slanje kratkih poruka i na broj 8888 šalje poruku sa tekstom " REG vecera ".

Očekivani rezultati:

Korisnik prima SMS poruku od programa i otvara je na svom mobilnom uređaju. Sadržaj poruke je „Registrovali ste se na: vecera. Ali нико trenutno ne pruza uslugu“. U bazi se beleži korisnikova registracija na željenu ključnu reč.

Sl.5.7. Izgled baze nakon testa 3 i testa 4

Test prošao:

DA.

7. 5.1.5 Test 5

Opis:

Ovaj test se oslanja na prethodne. Cilj ovog testa jeste da pokaže funkcionisanje obaveštavanja korisnika o bliskim uslugama. Nakon što se korisnik registrovao na informacije o "sok", sada će proći pored lokacije koja služi sok. U ovom konkretnom slučaju, kada je korisnik prošao pored lokacije dobija poruku : "10: Svatite kod nas na sok".

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka.

Koraci:

1. Akiviranje jednog korisnika u simulatoru NRG-a
2. Korisnik mobilnog uređaja pokreće aplikaciju za slanje kratkih poruka i na broj 8888 šalje poruku sa tekstom "REG sok".
3. Korisnik počinje da se kreće ka poziciji koja pruža uslugu "sok".

Sl. 5.8. Lokacija gde se nudi sok

Očekivano ponašanje:

Korisnik prima SMS poruku od programa i otvara je na svom mobilnom uređaju. Sadržaj poruke "10: Svatite kod nas na sok".

Sl. 5.9. Korisnik prolazi pored lokacije i prima poruku

Test prošao:

DA.

8. 5.1.6. Test 6

Opis:

U ovom testu korisnik se zadržava na jednom mestu (u blizini obavezne lokacije) duže od intervala između dva obaveštavanja (koji je 3 minuta u ovom slučaju). Cilj je da se proveri da li merenje vremena između dva intervala funkcioniše pravilno.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje jednog korisnika u simulatoru NRG-a
2. Korisnici počinju da se kreću ka pozicijama na kojima se nalaze lokacije sa obaveznim obaveštavanjem.
3. Zadržavanje korisnika u blizini lokacije od koje je primio obaveštenje duže od 3 minuta

Očekivano ponašanje:

Korisnik kada prođu 3 minuta prima novu poruku od programa sa jednakim sadržajem koji je dobio prvi put. Sadržaj poruke je opis usluga koje pruža lokacija

Sl. 5.10. Primljene poruke u razmaku od 2 minuta

Test prošao:

DA.

9. 5.1.7. Test 7

Opis:

Ovaj test proverava funkcionalnost u slučaju aktivnosti više korisnika. Aktiviraju se 3 korisnika i prolaze pored nekih obaveznih lokacija. Cilj je demonstracija rada u slučaju više korisnika.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje 3 korisnika u simulatoru NRG-a.
2. Pokretanje korisnika ka obaveznim lokacijama.

Očekivano ponašanje:

Kada se neki od korisnika se nađe blizini neke od obaveznih lokacija prima kratku poruku sa opisom usluga koje pruža ta lokacija. Takođe se podaci o korisnicima beleže u bazu podataka.

Sl. 5.11. Jedan korisnik dobija SMS Sl. 5.12. Drugi korisnik dobija SMS

Sl. 5.13. Izgled baze kada se aktiviraju 3 korisnika

Test prošao:

DA.

10. 5.1.8. Test 8

Opis:

Ovaj test proverava reakciju programa u slučaju kada korisnik pošalje loše formulisani poruku. Korisnik programu šalje SMS poruku sa sadržajem „REGsport“ , što je nepravilan format i kao rezultat korisnik dobija odgovor: „Pogresno formulisana poruka“.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje novoga korisnika
2. Korisnik mobilnog uređaja pokreće aplikaciju za slanje kratkih poruka i na broj 8888 šalje poruku sa tekstom "REGsport".

Očekivano ponašanje:

Korisnik prima kratku poruku sa sadržajem: „Pogresno formulisana poruka“ i njegova registracija se ne beleži u bazi.

Test prošao:

DA.

11. 5.1.9 Test 9

Opis:

Cilj ovoga testa jeste da se proveri reakcija kada programu stigne poruka od korisnika koji nema pravo na uslugu tj. ne nalazi se u predviđenom opsegu brojeva (6612300 - 6612309). Kao rezultat na poruku sa sadržajem: „REG rucak“ dobija se odgovor: „Niste korisnik ovog servisa“.

Preduslov:

Pokrenut poslužioc MySQL, simulator NRG-a i programsko rešenje zadatka. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje novog korisnika , ali sa brojem telefona van opsega, npr. 6339021
Sl. 5.14. Ubacivanje korisnika van opsega
2. Korisnik mobilnog uređaja pokreće aplikaciju za slanje kratkih poruka i na broj 8888 šalje poruku sa tekstom "REG rucak".

Očekivano ponašanje:

Korisnik prima kratku poruku sa sadržajem „Niste korisnik ovog servisa“ i njegov zahtev se ne beleži u bazu, kao ni informacije o njemu.

12. 5.1.10 Test 10

Opis:

U ovom testu se koristi isti postupak kao i u testu 2, ali sa razlikom u konfiguracionoj datoteci i tome da se program pokreće sa drugog računara na kome nije postavljen NRG, već je pokrenut na računaru sa IP adresom 192.168.30.54. Razlika u konfiguracionoj datoteci jeste to što su promenjena polja:

CosNamingResponder.0.host=192.168.30.54

url = jdbc:mysql:// 192.168.30.54:3306/

Kao rezultat dobijamo jednaku funkcionalnost kao i kada je program pokrenut na istom računaru na kome su simulator i poslužioc MySQL.

Preduslov:

Pokrenuti poslužioc MySQL i simulator NRG-a na računaru čija je IP adresa 192.168.30.54. Pokrenuto programsko rešenje na drugome računaru koji može da pristupi, preko mreže, računaru na kome su pokrenuti poslužioc MySQL i simulator NRG-a. Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Aktiviranje jednog korisnika u simulatoru NRG-a
2. Korisnika počinje da se kreće ka poziciji na kojoj se nalazi lokacija sa obaveznim obaveštavanjem.

Očekivano ponašanje:

Korisnik prima SMS poruku od programa i otvara je na svom mobilnom uređaju. Sadržaj poruke je opis usluga koje pruža lokaciju.

Test prošao:

DA

13. 5.1.11 Test 11

Opis:

U ovom testu se program pokreće bez prethodno pokrenutog simulatora NRG-a i kao rezultat u konzoli dobijamo informaciju o neuspešnom povezivanju na NRG.

Preduslov:

Pokrenut poslužioc MySQL. Baza podatak mora biti prazna pre pokretanja programskog rešenja. Simulator NRG-a mora biti ugašen.

Koraci:

1. Pokretanje programskog rešenja.

Očekivano ponašanje:

Program izbacuje informacije o grešci prilikom komunikacije sa NRG-om.

Sl. 5.15. Konzola kada simulator NRG-a nije aktivan

Test prošao:

DA.

14. 5.1.12. Test 12

Opis:

U ovom testu se program pokreće bez prethodno pokrenutog poslužioca MySQL i kao rezultat u konzoli dobijamo informacije o grešci.

Preduslov:

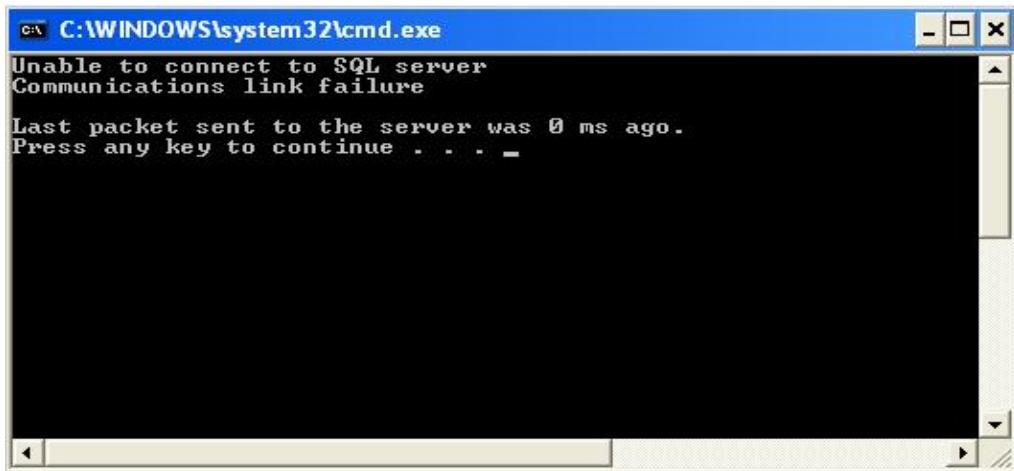
Pokrenut simulator NRG-a. Baza podatak mora biti prazna pre pokretanja programskog rešenja. Poslužioc MySQL mora biti ugašen.

Koraci:

1. Pokretanje programskog rešenja.

Očekivano ponašanje:

Program izbacuje informacije o grešci prilikom komunikacije sa poslužiocem MySQL.



Sl. 5.15. Konzola kada ne radi poslužioc SQL

Test prošao:

DA

13. 5.2 Testiranje sa alatom za automatsko testiranje (ATT)

Opis:

Testiranje sa ATT-om je odrđeno tako što je pokrenut ATT i posle toga program se pušta da radi određeno vreme. Tokom rada, na osnovu zahteva koje je program poslao NRG-u, ATT simulira situacije koje bi NRG trebao da prijavi programu. Te situacije su promena pozicije korisnike, promena statusa korisnika (aktivan ili ne aktivran) i slanje poruka na servisni broj. Pošto ATT ne može nasumično da generiše kratke poruka koje bi program mogao da prihvati kao ispravne, ovaj deo programa nije moguće verifikovati pomoću ATT-a. Praćenje pozicije i statusa korisnika, kao i obaveštavanje korisnika kratkom porukom kada se našao u blizini lokacije sa obavezim obaveštavanjem se može testirati i verifikovati korišćenjem ovog alata. Nakon 10 minuta program je i dalje uspešno funkcionisao, uz sve nasumične situacije koje je ATT simulirao.

Preduslov:

Pokrenut poslužioc MySQL i ATT. . Baza podatak mora biti prazna pre pokretanja programskog rešenja.

Koraci:

1. Pokretanje programskog rešenja.

Očekivano ponašanje:

Program bio trebao da obavlja sve predviđene funkcije za vreme rada ATT-a

Test prošao:

DA.

7. 6. Zaključak

Kao osnovni cilj ovoga zadatka bilo je upoznavanje sa Parlay/OSA spregom i razvijanje uluge koristeći mogućnosti koje pruža ona, i alati koji su vezani za nju. Parlay/OSA sprega tj. NRG se pokazao ka izuzetno olakšavajuća alatka za razvoj usluga za mobilnu telefoniju, jer sakriva svu

kompleksnost mobilne mreže koja se iza njega nalazi, a upotreba usluga mreže koje pruža NRG je jednostavna preko metoda određenih klasa. Kao dokaz ovoga se pruža prilično koristan program tj. usluga koji je razvijena korišćenjem ovih alata, za vrlo kratak period. Da bi program mogao komercijalno da se korisit bila bi potrebna dodatna dorada (omogućavanje dodavanja novih lokacija interaktivno, naplaćivanje, izrada korisničke grafičke sprege, ...), ali sva potrebna funkcionalnost je realizovana. Još jedno veliko ograničenje kod razvoja i optimizacije je bila loša i ograničena izvedba simulatora. Pored brojnih ograničenja u simulatoru se često javljaju greške pri radu i zastoji. Još jedna alatka koja je korišćena u razvoju jeste poslužioc MySQL koji se pokazao ka izuzetno moćna i korisna alatka i svojim mogućnostima je uspešno preuzeala deo dodane obrade koju su ograničenja simulatora prenela na program.

8. **7. Literatura**

- Ericsson Network Resource Gateway User Guide, *Ericsson AB 2005*
- Parlay/OSA Specification, *3GPP (3rd Generation Partnership Project) 2003*
- Ericsson H-OSA Interface Specification, *Ericsson Radio Systems AB 2003*
- Ericsson Network Resource Gateway Parlay/OSA US Statement Of Compliance, *Ericsson AB - 2004, 2005*
- Sam's Teach Yourself MySQL in 21 Days, Mark Maslakowski, 2000

8. **Prilog**

1. **8.1 Instalacija i pokretanje programa**

1. 8.1.1. Potrebni alati

Dve ključne komponente su potrebne da bi program mogao da se pokrene. To je simulator NRG-a (ili ATT) i poslužioc SQL. Program je testiran na poslužiocu MySQL 5.0. Pored poslužioca MySQL preporučljiva je i instalacija MySQL GUI Tool 5.0, grupe alata za lakše upravljanje poslužiocem MySQL. Nakon ovoga još je potrebno instalirati i Ericsson Network Resource Gateway. U sklopu ove instalacije, pored simulatora, primera, i alata za automatsko testiranje, dobija se i instalacija Java u potrebnoj verziji, a to je JAVA 1.3.1_15.

2. 8.1.2. Podešavanje okruženja

Nakon instalacije obaveznih alata potrebno je podesiti određene parametre. Kod podešavanja poslužioca MySQL najvažniji korak je kreiranje baze. Da bi se ovo olakšalo generisana je backup datoteka baze koja olakšava kreiranje baze, učitavanjem ove datoteke pomoću alata MySQL Administrator. Kada je baza stvorena, potrebno je podesiti korisnike MySQL baze i njihova prava nad bazama (u testiranju je korisnik nazvan Parlay i dozvoljena su mu sva prava nad bazom locationdetector). Takođe je potrebno i proveriti na kojem mrežnom prolazu će poslužioc MySQL da osluškuje upite.

Da bi se simulator uspešno pokrenuo posle instalacije, jedino što je potrebno jeste dodati u operativni sistem nove promenjive okruženja (environmental variable):

- **NRGSDK** promenjiva treba da sadrži putanju do direktorijuma u kome je instaliran SDK.

Npr.

NRGSDK=

C:\Program Files\Ericsson\Network Resource Gateway SDK\R5A02

- **ANT_HOME** je promenjiva koja sadrži putanju do direktorijuma gde je SDK instalirao Apache Ant koji koristi za pokretanje simulatora i ATT-a.

Npr.

ANT_HOME =

C:\Program Files\Ericsson\Network Resource Gateway SDK\R5A02\tools\ant

- **JAVA_HOME** je promenjiva koja sadrži putanju do direktorijuma gde je instalirana Java.

Npr.

JAVA_HOME=

C:\jdk1.3.1_15

I potrebno je osvežiti promenjivu **Path** koja već postoji i dodati:

%JAVA_HOME%\bin

Poslednja stavka koju je potrebno uraditi jeste iskopirati direktorijum sa programom i svim potrebnim datotekama na računar.

3. **8.1.3. Pokretanje programa**

Kada su svi parametri podešeni sledi pokretanje programa. Naravno da bi se pokrenuo program potrebno je prvo pokrenuti poslužioc MySQL. Primer kako se to radi je skripta *sql start.bat* :

```
cd "C:\Program Files\MySQL\MySQL Server 5.0\bin"  
mysqld-nt --console
```

Pored poslužioca MySQL potrebno je i pokrenuti simulator.

Pokretanje programa se obavlja pomoću skripte *run.bat* u kojoj je potrebno pravilno navesti putanje do JAVA biblioteka:

```
@ ECHO OFF  
java -classpath ".\classes\;.\classes\coreapi.jar;.\classes\utilityapi.jar;  
.classes\estapi.jar;.\classes\activation.jar;.\classes\mail.jar;  
.classes\corba_sun.jar;  
.classes\mysqlconnector-java-5.1.6-bin.jar;  
C:\Program Files\Java\jdk1.6.0_03\jre\lib\rt.jar;  
C:\Program Files\Java\jdk1.6.0_03\lib\dt.jar;  
C:\Program Files\Java\jdk1.6.0_03\lib\tools.jar" locationdetectLocationDetect
```