GPRS baziran merno-upravljački sistem

Uroš Pešović, Željko Jovanović, Siniša Ranđić

Sadržaj — **U ovom radu predstavljen je daljinski merno-upravljački sistem koji je zasnovan na Telit GM862-GPS modulima. Ovi modli koriste GPRS protokol za prenos izmrenih podataka ka serverskoj Web aplikaciji. Ukoliko je potrebno server je u stanju da pošalje upravljačke informacije sa kojima se koriguje rad uređaja (promena učestanosi slanja, resetovanje uređaja) ili upavlja aktuatorima koji su priključeni na uređaj. U radu je prikazan primer nadzora grupe udaljenih mernih stanica.**

Ključne reči — GPRS, Google Maps, Java, merno-upravljački sistem, Telit.

I. UVOD

UPRAVLJANJE na daljinu je oduvek bila aktuelna tema na području automatike. Prvi merno-upravljački sistemi, namenjeni za upravljanje udaljenim objektima, su koristili sopstvnu komunikacionu infrastrukturu, koja je uglavnom bila zasnovana na radio modemima. Vremenom, korišćenje ovakvih načina komunikacije postalo je nepraktično, pogotovu za manje aplikacije. Sveopšti trend u razvoju aplikacija ovog tipa, predstavlja korišćenje već postojeće infrastrukture kao što je mreža mobilne telefonije zajedno sa njemim protokolima za razmenu podataka WAP, GPRS, EDGE. Prednost ovakvog pristupa je mogućnosti da se uređaji mogu raditi bilo gde u svetu, nezavisno od izbora mrežnog operatera.

U ovom radu predstavljen je jedan takav sistem, koji se sastoji iz niza merno-upravljačkih uređaja i Web aplikacije koja se korsiti za prikupljanje i upravljanje radom uređaja. Pored ovog dela, Web aplikacija poseduje integrisan Google Maps geografski informacioni sistem , koji se koristi za prikaz položaja pojedinih uređaja na geografskoj karti.

II. MERNO-UPRAVLJAČKI UREĐAJI

Uređaji namenjeni za aplikacije ovog tipa mogu se klasifikovati na stacionarne i mobilne. Stacionarni uređaji ne menjaju svoju poziciju u toku vremena, tako da sa serverom vrše samo primopredaju mernih i upravljačkih veličina. Mobilni uređaji pored gorepomenutih veličina serveru šalju podatke i o svojoj trenutnoj poziciji koju dobijaju od strane integisanog GPS prijemnika.

U.Pešović, Ž. Jovanović, S. Ranđić, Tehnički Fakultet u Čačku, Univerzitet u Kragujevcu , Srbija (telefon: 381-32-302721, fax: 381-32-342101, e-mail: pesovic@yahoo.com, zjovanovici@gmail.com, rasin@tfc.kg.ac.rs).

Osnovu merno-upravljačkih uređaja čini modul GM862-GPS, italijanske kompanije Telit [1]. Pomenuti modul u sebi poseduju quad-band GSM/GPRS modem, koji koristeći mrežu mobilne telefonije prenosi izmerene vrednosti serveru. On takođe poseduju i 20-kanalni SiRFstarIII GPS prijemnik koji obezbeđuje preciznost određivanja pozicije, koja je u 90 procenata slučajeva tačnosti od 2.5m. Pored pomenutih funkcija na raspolaganju je i 13 digitalnih ulaza/izlaza koji se mogu koristiti za prikupljanje informacija o merenim veličinama. Pomenuti digitalni ulazi mogu se koristiti i za povezivanje sa digitalnim senzorima koji izmerene podatke šalju korišćenjem SPI ili I2C magistrale. Modul može raditi u dva režima rada: kao mikrokontroler ili kao modem. Ukoliko radi kao mikrokontroler, Telit modul koristi program napisan u Python script [2] programskom jeziku za urpavljanje radom celokupnog merno-upravljačkog uređaja. Ukoliko radi kao modem, upravljanje radom uređaja vrši se od stane eksternog mikrokontorlera ili mikro-računara. U tom slučaju Telit modul se kostiri za određivanje GPS pozicije i prenos podataka preko GPRS konekcije, pri čemu se modulom upravlja preko RS232 porta korišćenjem AT komandi. AT komande predstavljaju skup tekstualnih komandi pomoću kojih se kontrolišu modemski uređaji, kako bi se realizovale željene funkcije.

Komunikacija sa serverom se izvodi HTTP protokolom putem GPRS-a. Veza između uređaja i servera može se uspostaviti na dva načina pomoću *uplink* i *downlink* tipa veze.

Kod *uplink* veze uređaj je taj koji inicira komunikaciju sa web serverom i to samo kada ima potrebu za slanjem izmerenih podataka. Komunikacija započinje uspostavljanjem sesije sa klijentskom aplikacijom na web serveru i slanjem izmerenih podataka u vidu HTTP GET zahteva. Tom prilikom web server uređaju odgovara o uspešnom prijemu zahteva pomoću HTTP koda 200 OK. Ukoliko je potrebno, web server dinamički generiše neophodne upravljačke imformacije, koje se umeću u potvrdu o uspešno predatom zahtevu. Na ovaj način ostvaruje se indirektan prenos upravljačkih podataka između klijentske aplikacije na web serveru i uređaja. Prednost ovakog vida komunikacije ogleda se u jednostavnosti, kako na klijentskoj, tako i na strani uređaja. Veći deo vremena uređaj može provesti u neaktivnom stanju, što je jako korisno ako je uređaj baterijski napajan, što je čest slučaj sa mobinim uređajima. Uređaj po potrebi započinje komunikaciju sa web serverom za koju je dovoljno da poseduje privatnu IP adresu, što je i najčešći slučaj kod većine mobilnih operatera. U sledeća dva listinga pikazana je jedna sekvenca komunikacije između uređaja i klijentske aplikacije, pri čemu prvi listing predstavlja HTTP GET

zahtev koji uređaj upućuje klijentskoj aplikaciji, posle koga sledi njen odgovor.

```
Send data to: 91.187.132.16:8080
Socket: CONNECT
GET CSLGPS/GISRazvojnoOkruzenje/
NewPoint?Telit_id=3&lon=02021.1222E&
lat=4352.9047N&parm1=+7.2111&parm2=+7.2567&
parm3=+7.3125&parm4=+7.1000&Status=01
HTTP/1.0
```

```
Response: HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Length: 0
Date: Fri, 23 Jan 2009 19:11:19 GMT
Connection: close
```

Kod *downlink* komunikacije web server je taj koji inicira vezu sa nekim od uređaja. U tom slučaju server po potrebi vrši prozivanje uređaja od koga zahteva izmerene podatke. Ovaj način komunikacije je puno komplikovaniji od prethodnog jer zahteva da uređaj poseduje javnu IP adresu, kako bi mogao primiti zahtev sa servera. Takođe uređaj bi morao biti stalno uključen i u stanju da primi odgovarajući zahtev od strane servera.

Merno-upravljačkom sistem korišćen u ovoj aplikaciji je koncipiran na *uplink* tipu veze, koja se pokazala kao jednostavnija i svrsishodnija.

III. Web Aplikacija

Glavni cilj klijentskog dela projekta je razmena mernih i upravljačkih informacija sa uređajima kao i prezentovanje mernih podataka dobijenih od strane uređaja. Generisanje upravljačkih informacija bi se vršilo od strane korisnika ili automatski na osnovu vrednosti merenih parametara po ustaljenim pravilima za neki opseg merenih vrednosti.

Celokupno softversko rešenje je realizovano korišćenjem open source J2EE [3] tehnologije. Jedan od razloga zbog čega je izabrana J2EE tehnologija je i jednostavna mogućnost proširenja kako bi se pomenuti sistem koristio na mobilnim telefonima u vidu J2ME aplikacije. Realizovano softversko rešenje se može grubo podeliti na tri dela:

1. Deo za automatsko upravljanje koji i vrši razmenu podataka sa hardverom
2. Korisnički konktrolisan deo za upravljanje
3. Deo za prikaz rezultata klijentu

Deo softvera koji služi za razmenu podataka sa hardverom ne poseduje grafičku interpretaciju sa korisničke strane i izvršava se samo prilikom zahteva servletu koji uspostavlja uređaj koji prosleđuje podatke. Razvijen je u vidu jednog Java Servlet-a koga u određenim vremenskim intervalima proziva hardver koristeći HTTP GET zahtev. Na ovaj način uređaj predaje odgovarajuće podatke Servletu, koji vrši obradu istih i čuva ih u mySQL bazi podataka. Pri tom server uređaju potvrđuje uspešan prijem podataka pri čemu je u stanju da mu pošalje parametre pomoću kojih bi se korigovao rad uređaja.

Konkretno u ovom slanju paramatara se krije to automatsko upravljanje. Za svaki uređaj se tačno zna opseg dozvoljenih vrednosti kao i niz aktivnosti po kojima treba da se postupi ako je neki parametar van tog opsega. Mogućnosti upravaljnja su velike, od najprostijih komandi vezanih za rad telita, kao što su resetovanje uređaja, promena intervala primanja podataka, pa do upravljanja po nekom nizu aktivnosti da bi se vrednost nekog merenog parametra dovela u dozvoljeni opseg. Prilikom kreiranja HTTP zahteva od strane uređaja servlet vrši niz provera vezanih za konkretni uređaj kao i za vrednost parametra koji je prosleđen. Nakon toga kreira se odgovor u vidu jedne JSP stranice čiji sadržaj ustvari predstavlja niz upravljačkih komandi. Ukoliko je sadžaj generisane stranice prazan (HTTP kod 200 OK), ne treba vršiti nikavu korekciju rada uređaja. Ako na primer, stranica sadrži naredbu TIME_INTERVAL 30 onda to znači da će nakon toga uređaj promeniti svoj interval slanja podataka na 30 sekundi. Ovo može biti od velikog značaja ako je neki od paramatara u kritičnom opsegu pa se onda smanjem intervala dobija bolji uvid u njegovo dalje kretanje što omogućava i bržu reakciju. Kod ovakvog automatizovanog upravaljanja najvažnije je dobro poznavati sve uslove da bi se na osnovu iskustva u radu sa njima napisala univerzalna pravila koja bi bila uvek primenljiva, po kojima bi Servlet kreirao odgovor.

Korisnički kontrolisani deo za upravljanje radom uređaja je zamišljen tako da se za izbrani uređaj prosledi komanda koja će se izvršiti ne vezano za vrednosti parametara. Jednostavno korisnik će promeniti neki od parametara uređaja kao što je promena vremenskog intervala za slanje izmerenih podataka ili pak resetovati uređaj. Direktno slanje upravljačkih informacija nije moguće već se komanda može poslati uređaju prilikom sledeće primipredaje izmerenih podataka. Tom prilikom servlet proverava se da li postoji spremna komanda za željeni uređaj i ukoliko ona postoji kreira se JSP strana čiji sadržaj predstavlja željenu komandu.

Pošto se servletu može pristupiti pomoću HTTP zahteva sa bilo kog mesta u svetu ovaj način komunikacije daje jednu globalnost celoj realizaciji projekta i mogućnost korišćenja iz bilo kog mesta na svetu gde postoji pristup internet servisima. Ova navedena osobina u realizaciji komunikacije pored te velike prednosti ima i jednu manu tj. otvorenost ka napadima i simuliranju od strane nekih drugih lica koji bi želeli da lažiraju vrednosti prosleđenih parametara. Zbog toga je vema važno obratiti posebnu pažnju na zaštitu i proveru validnosti uređaja koji šalju podatke. Jedan od prvih načina zaštite je dodeljivanje uređaju jednstvenog identifikacionog broja koji bi se proveravao pri svakom prijemu podataka. U ovom polju ima još mogućnosti za poboljšanje sigurnosti eventualno korišćenjem određenih kriptografskoh metoda čime bi se očuvalo interitet i verodostojnost podataka.

Poslednji delo aplikacije zadužen je za prikaz prikupljenih podataka korisniku. Pošto se Telit uređaji poziciono identifikuju slanjem svojih kordinata tj. geografske širine i dužine za podršku je izabran Google Maps GIS koji na veoma lak i funkcionalan način daje geografsku reprezentaciju praćenih objekata što u sistemima sa više uređaja ima ogromnu prednost.

Jednostavno u svakom trenutku je moguće imati grafički prikaz vrednosti parametara na uređaja klikom na marker. Google Maps API [4] daje velike mogućnosti zahvaljujući celokupnoj pokrivenosti zemljine kugle sa satelitskim i aero-foto snimcima visoke rezolucije. Takođe, veoma dobro je razvijen maping sistem koji je u zemljama Zapadne Evrope i Americi razvijen do te mere da su ucrtane ulice sa svojim nazivima u skoro svakom naseljenom mestu. Srbija je za sada pokrivena samo mrežom autoputeva i magistralnh puteva, ali u bliskoj budućnosti očekuje se kompletna GIS pokrivenost.

Princip rada Google Maps API-ja je da se kompletan GIS sistem nalazi na Google-ovom serveru. Korisnik prosleđuje koordinate i parametre za prikaz, dok server odgovara šaljući zahtevani grafički saržaj. Jedini uslov koji mora da se ispuni da bi moglo da se radi sa Google maps API-jem je da na svakoj Web stranici (u našem slučaju na JSP stranici) na kojoj treba da se prikaže mapa mora biti importovan *script tag* u okviru *head taga* HTML (JSP) stranice sa upisanim *key*-om koji se dobija od strane Google-a. Ključ se dobija besplatno i samo je potrebno uneti *URL* adresu Web servera na kojem će Google Maps biti korišćen. Google će generisati jedinstvenu ključ (*key*) i poslati je na navedeni Google mail nalog. Ključ (*key*) je potrebno postaviti na sledeći način na prethodno opisanu poziciju.

```
<script src=http://maps.google.com/maps?file=api&amp;
v=2.x&amp;
key=ABQIAAAA6HHnyYiT2EVEKFG_RERksh
QqcwwaKzD9OpoV4WAgC4lSGJ_ZoxSrievm
QkroThPpsog3YNQBW33JsQ
type="text/javascript"></script>
```

Nakon toga može se pristupiti prikazivanju slike sa željenim koordinatama. Prvo je potrebno definisati u kojem elementu u okviru HTML stranice se želi prikazati slika i to u okviru javascripta koji će se učitavati pri učitavanju stranice.

Google Maps poseduje tri tipa prikaza mapa:

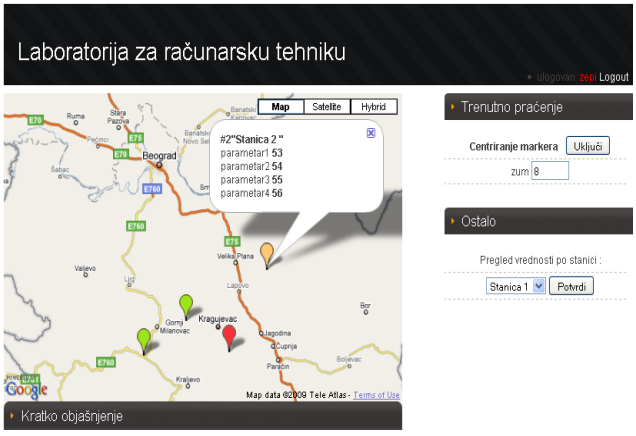
- ***map*** - predstavlja geografsko-informacionu podlogu mape sa ucrtanim putevima i nazivima ulica
- ***satellite*** - predstavlja satelitki ili aero-foto snimak željene lokacije
- ***hybrid*** - predsatvlja kombinaciju prethodna dva tipa

Prilikom prikazivanja željene lokacije potrebno je serveru proslediti tip mape i koordinate lokacije koju želimo da nam bude prikazana. Koordinate prikazane lokacije odgovaraju centru slike. Pored toga korisnik na raspolaganju ima mogućnost naknadnog izbora koji tip mape želi da bude prikazan. Jasnoća mapa zavisi od kvaliteta snimaka. Pojedine lokacije od većeg značaja poseduju veoma kvalitetne snimke sa mogućnošću uvećanja čak do 17 puta. Mapu je moguće uvećavati i bez pomoćnih kontrola levim duplim klikom miša, a smanjivati sa desnim duplim klikom miša. Važno je napomenuti da Google Maps besplatna licenca omogućava izvršenje do 500 000 zahteva u toku dana. Ako je potreban veći broj zahteva za slučajeve da se nadgleda veliki broj uređaja treba se obratiti Googlu Maps-u sa zahtevom za povećanje tog broja.

Sama J2EE platforma i Google Maps nisu mogli da zadovolje sve potrebne zahteve pa je pri realizaciji korišćen i JavaScript jezik u kombinaciji sa AJAX [5] tehnologijom. Razlog za to je što je potrebna dinamika isčitavanja podataka iz baze koji se menjaju nakon određenog vremena tj. posle svakog slanja podataka od strane hardvera. Jednostavno na ovaj način se omogućava praćenje željenih parametre za željenu stanicu bez ikakve akcije korisnika.

Kao kontejner korišćen je Apache Tomcat verzije 6.0.16. Razlog korišćenja Apache Tomcat Applications server-a u odnosu na druge varijante Jboss, Jrun, Glashfish Applications Servers itd. je veoma jednostavan način povezivanja sa razvojnim alatom Eclipse koji je korišćen prilikom izrade.

Projekat je realizovan kao Web aplikacija koja se nalazi na serveru Labaratorije za Računarsku tehniku [6], Tehničkog Fakulteta u Čačku. Aplikacija sastoji se iz dve celine. Jedna omogućava trenutno praćenje više stanica, tj. prikaz njihovog polažaja u realnom vremenu sa informacijama o geografskoj širini i dužini kao i o vrednostima parametara vezanih za svaku od stanica što je i prikazano na Slici 1. Pošto ovaj sistem ima ulogu u nadgledanju vrednosti parametara, markeri kojima se predstavlja stanica na mapi mogu biti zelene, žute ili crvene boje u zavisnosti od vrednosti parametara tj. da li su oni u dozvoljenom, kritičnom ili ne dozvoljenom opsegu respektivno. Što se tiče konkretne funkcionalnosti, status uređaja vidi se na osnovu boje markera a vrednost svih parametara se dobojaju klikom na marker. Na ovaj način se dobija celokupna preglednost sistema sa mogućnošću pregleda svih parametara na jednoj stanici uz vizualni prikaz njihovog geografskog položaja.



Slika 1. Pregled trenutnog praćenja stanica

Pored toga postoji i mogućnost pregleda vrednosti za određenu stanicu u toku željenog vremenskog intervala. Prilikom pregleda dobija se tabelarni prikaz vrednosti uz vizuelnu pomoć u vidu boja koje predstavljaju status određene stanice u datom trenutku. Na ovaj način se može pregledati kretanje parametara i proveriti da nije propušteno nešto od važnih dešavanja što možda nije detektovano okom zbog brzine reakcije servera ili možda

zbog odsutnosti osobe koja nadgleda sistem u trenutku nekog prekoračenja.

Veoma važna tema za diskutovanje je i koja količina obrade podataka treba da ostane na uređaju a koja na serveru gde je postavljena aplikacija. Pri obradi na uređaju pored podataka slao bi se još jedan parametar koji bi predstavljao njegov status. U slučaju obrade na serveru za n stanica na kojima se prati m parametara predstavljalo bi $m \cdot n$ obrada na svakih nekoliko sekundi na koliko bi se radilo ponovno proveravanje. Ovakva obrada bi mogla dosta da optereti server za više posmatranih stanica ali na serveru bi se obrada radila mnogo lakše, uz pomoć baza podataka kao i svih mogućnosti koje donosi jedan jezik visokog nivoa kao što je Java. Po ovom pitanju Pyton je dosta oskudniji. Ova tematika će i dalje biti predmet istraživanja a najverovatnije je da je neki kompromis između ova dva rešenja i najoptimalnije rešenje.

LITERATURA

[1] *GM862-GPS Hardware User Guide*, Telit Communications S.p.A., 1vv0300728 Rev. 8 -2007

[2] E*asy Script in Python*, Telit Communications S.p.A., 80000ST10020a Rev.7 - 19/12/07

[3] Paul Perrone Venkata S.R. Tom Schwenk, J2EE Developer's Handbook**,** 2003

[4] http://code.google.com/apis/maps

[5] Kris Hadlock, Ajax for Web Application Developers (Developer's Library) , 2006

[6] http://csl.tfc.kg.ac.yu:8080/GISRazvojnoOkruzenje/

ABSTRACT

Abstract – **In this paper, we presented the remote monitoring and control system based on Telit GM862-GPS modules. These modules use GPRS protocol for transfer of measured data to server Web application. If necessary, server is capable to send required control information to the device in order to control it (change of frequency, reset the device) or controll actuators that are connected to the device. The paper presents an example of remote stations monitoring.**

GPRS BASED REMOTE MONITORING AND CONTORL SYSTEM

Uroš Pešović, Željko Jovanović, Siniša Ranđić