# Web Servis i klijentska GIS aplikacija za akviziciju parametara sa udaljenih stanica

Željko Jovanović, Uroš Pešović, Siniša Ranđić

Sadržaj — **Mogućnost praćenja promena vrednosti mernih veličina u realnom vremenu sa udaljenih lokacija može biti od velike koristi. Kompletna realizacija rešenja putem Web Servisa daje jedan veći nivo u globalizaciji celog projekta. Poznavanje struktura Web Servisa omogućuje klijentskim aplikacijama da komuniciraju sa njim koristeći SOAP (Simple Object Access Protocol) protokol koji je ne zavistan od platforme i predstavlja HTTP protokol kojim se šalju XML (eXtensible Markup Language) fajlovi. Web servis pruža informacije putem publikovanog WSDL (Web Service Description Language) fajla koje se korisniku prezentuju putem klijentske aplikacije. U ovom radu prikazana je akvizicija merenih veličina sa udaljenih lokacija pomoću Web Servisa. Kompletno softversko rešenje je realizovano korišćenjem open source J2EE tehnologije. Za realizaciju ovog rešenja bilo je moguće koristiti i druge programske jezike ali Java izabrana zbog velikog broja funkcija koje pruža JDK (Java Development Kit). Za geografsku podlogu iskorišćen je Google Maps, globalni geografski informacioni sistem (GIS), koji je u formi Google Maps API-ja integrisan u našu Web aplikaciju. GIS predstavlja sistem za upravljanje prostornim podacima i prikaz geografskih inoformacija a preciznost podataka i pokrivenost celog sveta konstantno se unapređuje.**

**Ključne reči — Web Servis, SOAP, WSDL, UDDI, GPRS, Google Maps, Java.**

## I. Uvod

SVE veća zastupljenost Interneta pruža mogućnost distribucije podataka putem Web Servisa. Ovaj način distribucije informacija ima prednost da se one putem klijentskih aplikacija korisniku prezentuju na odgovarajući način. Ovakvi izvori informacija su sve zastupljeniji a kada se korisnici prilagode ovakvom načinu pribavljanja i distribucije informacija teško da će moći da se odviknu od njih. Era Web Servisa tek dolazi. U ovom radu će biti opisan jedan Web Servis i jedna aplikacija zasnovana na podacima koje dobija od servisa a kao podršku koristi Google Maps. Geografske informacioni sistemi se neprekidno usavršavaju i osvežavanju sa ažurnim podacima. Pojavom Google Maps-a, globalnog geografskog informacionog sitema, stekle su se mogućnosti da takav jedan sistem bude primenljiv bilo gde na svetu u formi prilagođenoj specifičnim potrebama korisnika.

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

## II. Web Servisi

W3C konzorcijum je 2000. godine započeo sa razvojem **SOAP** ( Simple Object Access Protocol ) specifikacije. Ideja je bila da podaci koji se prenose između različitih komponenti serijalizuju pomoću XML (eXtensible Markup Language) jezika, transportuju i zatim deserijalizuju u format koji je pogodan za odredišnu platformu. Tokom narednih godina, W3C je objavio **WSDL** (Web Service Description Language) specifikaciju. Ovaj standard, zasnovan na XML-u, je nudio jezik za opis interfejsa Web Servisa na standardizovan način - način na koji Web Servis prezentuje ulazne i izlazne parametre poziva, strukturu funkcije, prirodu poziva (samo ulazna, ulazna/izlazna …).

Kompletiranje prve generacije standarda Web Servisa je učinjeno sa **UDDI** (Universal Description, Discovery and Integratio) specifikacijom. Ova specifikacija dozvoljava kreiranje standardizovanih registara deskriptora servisa kako u okviru neke organizacije, tako i van njenih granica. UDDI nudi mogućnost da Web Servisi budu registrovani na jednoj centralnoj lokaciji, gde ih zatim mogu otkriti potraživači servisa - pretražujuci registre po imenu, identitetu, kategorijama ili po specifikaciji koju obezbeđuje Web Servis.

Veza između ova tri tehnologije [1], [2] (SOAP, WSDL, UDDI) može da se opiše na sledeći način: aplikacija koja predstavlja klijenta Web Servisa treba da locira drugu aplikaciju ili deo poslovne logike koji se nalazi negde na Web-u. Klijent pretražuje UDDI registar da bi pronašao servis, bilo po imenu, kategoriji, identifikatoru ili podržanoj specifikaciji. Kada ge pronađe klijent uzima informaciju o lokaciji WSDL dokumenta iz UDDI registra. WSDL dokument sadrži informacije o tome kako kontaktirati Web sevis, i informacije o formatu za poruke, najčešće u XML shemi. Klijent pravi SOAP poruku u skladu sa XML shemom koju je našao u WSDL-u i šalje zahtev računaru na kome se nalazi servis. (Web Servis može pristupati i enkapsulirati druge Web Servise da bi izvrsio svoju funkciju).

Trenutno za izradu Web Servisa dominiraju J2EE i .NET platforma. Odabir na kojoj platformi će se raditi zavisi samo od subjektivnih faktora ili zahteva klijenata. I jedna i druga platforma omugućavaju potpunu funkcionalnost.

Web Servis je izrađen na J2EE platformi [3], [4], [5] u cilju demonstracije realizacije kompletnog rada na jednoj platformi. Ideja je bila da se napravi Web Servis koji bi pružao geografske i metereološke podatke o gradovoma u Srbiji u realnom vremenu. Za potrebe čuvanja podataka napravljna je baza podataka u MySQL-u u koju se upisuju podaci prilikom svakog merenja. Sa ovom bazom podataka konkretno radi Web Servis i obrađuje podatke koje iz nje dobija. Funkcionalnost Web Servisa se trenutno ogleda u tome da za ime grada vraća najnovije parametre za taj grad i to:

- Geografsku širinu – latitude
- Geografsku dužinu – longitude
- Temperaturu vazduha– air temperature
- Vazdušni pritisak – air pressure
- Vlažnost vazduha – air humidity

Za realizaciju Web Servisa na J2EE platformi Eclipse Ganimade je korišćen kao razvojno okruženje sa instaliranim dodacima za rad sa Web Servisima. Za ceo postupak objavljivanja Web Servisa potrebno kreirati WSDL fajl koji opisuje šta treba da radi taj Web Servis. Nakon toga se kreira UDDI fajl kojim se objavljuje Web Servis tj. upisuje se u jedinstveni registar servisa koji omogućava da servis bude pronadjen sa bilo koje lokacije u svetu. Poslednji deo je deo za komunikaciju putem SOAP protokola, XML fajla koji se šale HTTP protokolom. Ovakava način komunikacije omogučuje platformsku i jezičku ne zavisnost ali ga je zato potrebno izparsirati tj. izdvojiti korisne informacije iz njega u formatu koji odgovara programu. Pri realizaciji ova tri koraka korišćen je Apache AXIS framework. AXIS (Apache Extensible Interaction System) predstavlja procesor SOAP poruka i omogućava lako parsiranje i izdvajanje korisnih informacija. Uloga AXISA-a je u prenosu podataka između transportnog sektora i biznis sloja aplikacije. U okviru dinamičkog web projekta kreiranog sa ulogom da se od neke njegove klase napravi projekat u lib direktorijumu moraju biti importovani sledeći jar fajlovi koji daju punu funkcionalnost AXIS framework-u:

*commons-discovery-0.2.jar, commons-logging.jar, jaxrpc.jar, saaj.jar, wsdl4j.jar, axis.jar*

Kada se kreira klasa u kojoj se nalaze samo metode koje će predstavljati funkcionalnost Web Servisa onda se ta klasa publikuje u Web Servis. Kontrolu nad ovim delom ima opisani AXIS framework koji kreira XML parser za odgovarajuće zahteve koje treba servis da obrađuje. Takođe kreira se i odgovarajući WSDL fajl u kome se da je opis metoda kao i lokacija na kojoj će biti publikovan Web Servis. Trenutno Web Servis je portiran na serveru Labaratorije za Računarsku tehniku Tehničkog fakulteta u Čačku na sledećoj URL adresi:

http://csl.tfc.kg.ac.rs:8080/WebService/services/WS

Kompletne informacije u strukturi servisa dobijaju se dodavanjem ***?wsdl*** na kraj prethodno pomenutog url-a tj:

http://csl.tfc.kg.ac.rs:8080/WebService/services/WS?wsdl

dobija se izgled WSDL fajla na osnovu koga se može videti opis metoda. Posebno važni delovi koda koji se dobijaju odlaskom na navedeni link prikazani su ispod:

- opis klase koju Web Servis vraća:

```
<complexType name="Podaci">
<sequence>
   <element name="latitude" nillable="true"
   type="xsd:string"/>
   <element name="longitude" nillable="true"
   type="xsd:string"/>
   <element name="naziv" nillable="true"
   type="xsd:string"/>
   <element name="pritisak" nillable="true"
   type="xsd:string"/>
   <element name="temperatura" nillable="true"
   type="xsd:string"/>
   <element name="vl_vaz" nillable="true"
   type="xsd:string"/>
</sequence>
</complexType>
```

- opis metode koja vraća klasu Podaci

```
<wsdl:portType name="WS">
   <wsdl:operation name="getData">
      <wsdl:input  message="impl:getDataRequest"
      name="getDataRequest"/>
          <wsdl:output    message="impl:getDataResponse"
          name="getDataResponse"/>
   </wsdl:operation>
</wsdl:portType>
```

- naziv servisa kao i njgova lokacija

```
<wsdl:service name="WSService">
<wsdl:port binding="impl:WSSoapBinding"
name="WS">
<wsdlsoap:address location=
"http://csl.tfc.kg.ac.rs:8080/WebService/services/WS"/>
```

## III. Web Aplikacija

Za svrhe testiranja Web Servisa kreirana je jedna aplikacija na J2EE platformi koja uz pomoć Google Maps-a prikazuje grad na mapi označen markerom. Klikom na marker se vrednosti parametara postaju vidljive.

Web Servis kreiran na J2EE platformi kreira i test klijent projekat koji spakovan u jar arhivu može olakšati implementaciju funkcionalnosti Web Servisa klijentima na J2EE platformi bez čitanja WSDL fajla i strukture podataka koje servis zahteva za rad. Implementacija Web Servisa u J2EE klijentsku aplikaciju lako se postiže sledećim kodom :

```
WSServiceLocator locator= new
WSServiceLocator();
   WS ws;
   Podaci p = new Podaci();
   ws = locator.getWS();
   p = ws.getData(grad);
```

Nakon ovoga u instanci klase Podaci p se nalazi ceo objekat čiji se atributi mogu prezentovati na željeni način. Drugi način je da se na osnovu WSDL fajla na klijentskoj strani kreira klasa Podaci koja će odgovarati opisu iz WSDL fajla. U primerak ove klase smešta se odgovor na zahtev koji se kreira ka URL-u Web Servisa.

Rad sa Web Servsom preko klijentskih aplikacija izrađenih na .NET platformi zahteva kreiranje reference (Service Reference) u kojoj se naznači URL Web Servisa. Nakon toga se kroz metode kreiranog primerka klase Web Servis reference može raditi sa podacima na željeni način. Primer pristupu parametru pritisak za grad Niš prikazan je sledećim kodom:

```
reference .WSService objPodaci = new
reference.WSService();
objPodaci.getData("Nis").pritisak;
```

Sama J2EE platforma nije mogla da zadovolji sve potrebne zahteve pa je pri realizaciji korišćen i JavaScript jezik. Iz pomenute platforme za rad su korišćene JSP (Java Server Pages) strane za reprezentovanje obrađenih zahteva. One su realizovane korišćenjem Expression Language-a, koji pruža veoma jednostavan pristup podacima sačuvanim u JavaBeans komponentama. 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.

Kao geografska podrška pri realizaciji projekta korišćen je Google Maps API [6] koji 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. Naša zemlja je za sada pokrivena samo mrežom autoputeva i magistralnh puteva i važnijih ulica, ali u bliskoj budućnosti očekuje se kompletna GIS pokrivenost. Princip rada Google Maps API-ja je da se kompletan GIS sitem nalazi na Google-ovom serveru. Korisnik prosleđuje koordinate i parametre za prikaz, dok server odgovara prosledjujuci zahtevani grafički saržaj. Glavni 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. U njemu će biti upisan *key* koji se dobija od strane Google-a. Ključ se dobija besplatno pod uslovom da postoji otvoren nalog (mail) na Google-u. Da bi se dobio *key* 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 što će omogućiti prikaz slike željenih koordinata.

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

Takođe potrebno je definisati u kojem elementu u okviru HTML stranice se želi prikazati slika i to u okviru javascripta koji će se učitavati pri učitavanju stranice:

```
if (GBrowserIsCompatible()) {
        var map = new GMap2
(document.getElementById("map_canvas"), {
        size :new GSize(570, 380)});
```

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.

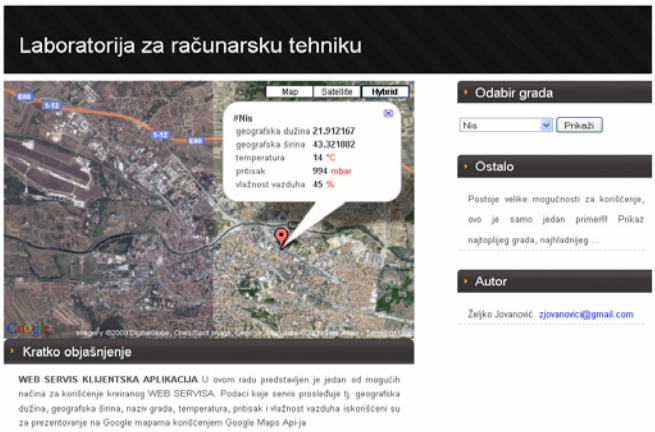
```
map.setMapType(G_SATELLITE_MAP);
map.setCenter(new GLatLng(latitude,longitude),13];
```

Za prikaz Google mapa na jsp stranici potrebno je uneti ključ koji se dobija od Googla i važi za jednu URL adresu. Nakon toga se može raditi sa opcijama koje nudi Google Maps uz pomoć Javascript jezika.

Klijentska test aplikacija [7] se može trenutno pogledati na sledećoj URL adresi:

http://csl.tfc.kg.ac.rs:8080/WSClientTest/

Na slici 1 je prikazan izgled aplikacije sa parametarima za grad Niš.




*Slika 1. Izgled test aplikacije sa prikazom parametara*

Naravno ovi parametri se mogu iskoristiti i na mnoge druge načine tj. u zavisnosti od potreba klijenata. Ovo je još jedna veoma dobra osobina Web Servisa ali i logike distribuirane obrade podataka.

Trenutno je u fazi razvoja proširenje funkcionalnosti Web Servisa sa metodama koje bi pružale listu Podataka. Ovaj koncept bi omogućio da se u jednom trenutku dobiju parametri za više gradova (stanica). Takođe i proširenje funkcionalnosti Web Servisa sa atributima koji bi signalizirali ne poželjna stanja parametara omogućilo bi i prezentovanje markerima različitih boja. Ovakav prikaz bi korisnicima pružao podatke i pre klika na marker na slici jer bi na primer crveni marker označavao da na tom mernom mestu neki parametar je van predviđenih granica.

## IV. ZAKLJUČAK

Pošto je projekat u fazi razvoja, predstoji još puno rada prvenstveno na povećanju pouzdanosti i bezbednosti primopredaje podataka. Normalno postoji još mnogo ideja koje bi mogle biti sprovedene sa već postojećom arhitekturom projekta. Razvoj aplikacije u više slojnoj arhitekturi amogućava lako proširivanje koda i dodavanje metoda Web Servisu što bi uvećalo njegovu funkcionalnost. Konkretno moguće je dodati metode za prikaz najtoplijeg ili najhladnijeg grada, pronalaženje rastojanja između dva grada sa svim razlika u metereološkim parametrima između njih što bi bilo veoma interesantno kada se želi putovati iz jednog u drugi grad.

## LITERATURA


[1] Cardoso J., Sheth A.P. *Semantic Web Services, Processes and Applications*, Springer 2006

[2] G. Alonoso, F. Casati, H. Kuno, V. Machiraju *Web Services, Concepts, Architectures and Applications*, Springer,2003.

[3] Steve Graham, Simeon Simeonov, Toufic Boubez, Doug Davis, Glen Daniels, Yuichi Nakamura, Ryo Neyama, *Building Web Services with Java™: Making Sense of XML, SOAP, WSDL, and UDDI*, Sams Publishing, december 2001

[4] Antonio Pintus, *Building Web Services with Java*, version 1.2

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

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

[7] http://csl.tfc.kg.ac.rs:8080/WSClientTest/



## ABSTRACT

*Abstract* – **Ability to remotely monitor value changes in real time can be of great benefit. Complete realization of solutions in the form of a Web Service provides one more level in the field of globalization. When the structure of the service are known client application communicates with the web service using SOAP (Simple Object Access Protocol) protocol which is independent of platform and represents the HTTP protocol which send the XML (eXtensible Markup Language) files. Web Service is offering information which are presented to the final user by client applications by published WSDL (Web Service Description Language) file. This article is striving to do just that and show how it would be possible to remotely monitor measured values from different locations using Web Service. The entire software solution has been realized using the open source J2EE technology. It is possible to use other programming language, but because of the large number of functions that provides JDK (Java Development Kit) Java programming language was selected. For geographic layer, I use Google Maps, global geographic informational system GIS, which is used in form of Google Maps API in my Web application. GIS is a system for managing spatial data and displaying geographical information, and accuracy and coverage around the world are constantly improving.**


## WEB SERVICE AND CLIENT GIS APPLICATION FOR DATA ACQUISITION FROM REMOTE STATIONS


Željko Jovanović, Uroš Pešović, Siniša Ranđić