SOAP protokol i njegova primjena

Balentović, Bruno

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELTEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

SOAP PROTOKOL I NJEGOVA PRIMJENA

Završni rad

Bruno Balentović

Osijek, 2018

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Osijek, 18.09.2018.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Ime i prezime studenta:	Bruno Balentović
Studij, smjer:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI4455, 21.09.2017.
OIB studenta:	88635203886
Mentor:	Mr.sc. Anđelko Lišnjić
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Izv. prof. dr. sc. Vanja Mandrić-Radivojević
Član Povjerenstva:	Izv. prof.dr.sc. Slavko Rupčić
Naslov završnog rada:	SOAP protokol i njegova primjena
Znanstvena grana rada:	**Telekomunikacije i informatika (zn. polje elektrotehnika)**
Zadatak završnog rada	Simple Object Access Protocol je specifikacija protokola za razmjenu poruka za razmjenu strukturiranih informacija u implementaciji web usluga u računalnim mrežamai jedno od obilježja je neutralnost i neovisnost o operativnim sustavima. U zadatku je potrebno detaljno obraditi SOAP protokol, objasniti razlog nastanka i povijest, navesti njegove osnovne karatkeristike i značajke. Pojasniti način rada u web servisima. Navesti i objasniti što je potrebno za uspješno korištenje SOAP protokola.
Prijedlog ocjene pismenog dijela	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	18.09.2018.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 29.09.2018.

Ime i prezime studenta:	Bruno Balentović
Studij:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI4455, 21.09.2017.
Ephorus podudaranje [%]:	2

Ovom izjavom izjavljujem da je rad pod nazivom**: SOAP protokol i njegova primjena**

izrađen pod vodstvom mentora Mr.sc. Anđelko Lišnjić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Iz ideje da se olakša međuaplikacijska komunikacija, da se smanje zahtjevi na implementaciju protokola i omogući međuprotokolna kompatibilnost nastao je SOAP protokol (*Simple Object Access Protocol*). Protokol nije ovisan o platformi, kao ni o programskom jeziku i operacijskom sustavu. [1] Temelji se na XML jeziku i omogućava razmjenu podataka među necentraliziranim aplikacijama i sustavima putem drugih transportnih protkola, među kojima je najkorišteniji HTTP zbog mogućnosti sigurnog prolaska kroz vatrozid (engl. *Firewall*).

SOAP protokol predstavlja osnovni komunikacijski protokol namjenjen razmjeni tekstualnih poruka u web servisima. Njime je opisan način na koji će poruka biti oblikovana prilikom prijenosa nekim od transportnih protokola te način na koji će ista poruka biti razmjenjena i obrađena među aplikacijama. Ta se poruka još naziva i SOAP XML dokument ili SOAP omotnica (engl. *envelope*), a može sadržavati uputstva o obradi poruke, prava pristupa poruci, sigurnosna proširenja, poruke o pogrešci i tekst poruke sa korisnom informacijom (engl.*payload*). [2]

SOAP protokolom moguće je slati zahtjeve i primati odgovore pisane XML jezikom. Najčešće se koristi za izmjene transakcija, koordinaciju, sigurnost te pouzdanost u komunikaciji. Moguća je i jednosmjerna i dvosmjerna komunikacija. Za jednosmjernu komunikaciju karakteristično je što nema povratnog odgovora, dok u dvosmjernoj komunikaciji slijedi odgovor na zahtjev. [1]

Prvi dio ovog rada opisuje razvoj SOAP protokola, kako je nastao te kako se razvijao kroz povijest do danas. U drugom dijelu se pojašnjavaju i navode ključne karakteristike SOAP protokola, te građa SOAP poruke odnosno omotnice i njezinih dijelova. Treći dio opisuje osnovni princip rada i način primjene protokola. U praktičnom dijelu rada na jednostavnom primjeru prikazan je rad SOAP protokola kroz zahtjev i odgovor. Na kraju rada dana je usporedba s alternativnim rješenjem, uporabom REST-a (*Representational State Transfer*) u web servisima.

1.1. Zadatak završnog rada

Zadatak završnog rada je objasniti što je SOAP protokol, objasniti razlog nastanka i povijest, navesti njegove osnovne karatkeristike i značajke. Pojasniti način rada u web servisima. Navesti i objasniti što je potrebno za uspješno korištenje SOAP protokola. Pokazati način uporabe na jednostavnim primjerima. Te ga usporediti sa alternativnim rješenjem.

2. RAZVOJ SOAP PROTOKOLA

SOAP protokol kreirali su i dizajnirali Microsoftovi inženjeri Bob Atkinson, Mohsen Al-Ghosein, Don Box i Dave Winer 1998. godine s ciljem da osiguraju veću sigurnost u komunikaciji među aplikacijama te smanje probleme sa međuprotkolonom kompatibilnosti.[3] Nastao kao zamjena za do tada korištene CORBA (*Common Object Request Broker Architecture*) i DCOM (*Distributed Component Object Model*) standarde koji su koristili pozive udaljenih procedura (engl. *Remote Procedure Calls*), a komunikacija se odvijala binarnim porukama, što dovodi do problema sa sigurnosti. Stoga je bilo potrebno zamijeniti navedene standarde jer HTTP nije namjenjen za binarnu komunikaciju, već tekstualnu komunikaciju koju koristi SOAP protokol. Specifikacije prve verzije dostupne su tek kada su poslane *IETF*-u u rujnu 1999. i nisu postale standardom . Verzija 1.1 objavljena je u svibnju 2000. godine, ali također nije odobrena kao standard. U početku kratica SOAP značila je za *Simple Object Access Protocol*, no pojavom verzije 1.2 kratica postaje službeno ime protokola. Tek u lipnju 2003. godine verzija 1.2 sa svojim specifikacijama postaje standard nakon odobrenja W3C organizacije. Postaje temeljni sloj složenim web servisima koji se temelje na XML-u, WSDL-u i UDDI-u. Do srpnja 2009. godine postojao je tim koji je brinuo za održavanje protokola pod nazivom *W3C XML Protocol Working Group.* Do danas korištenje SOAP protokola sve je manje zbog boljih specifikacija alternativnog rješenja kao što je REST. [4] No pretpostavka je da se SOAP nebi trebao zamijeniti REST-om još neko vrijeme zbog sigurnosnih proširenja te mogućnosti rada na više različitih transportnih protokola.

3. OSNOVNE KARAKTERISTIKE

Protokol se zasniva na XML-u (*EXtensible Markup Language*) i može raditi na bilo kojem transportnom protokolu kao što su na primjer HTTP (*Hyper Text Transport Protocol*), SMTP (*Simple Mail Transport Protocol*), TCP (*Transport Control Protocol*), UDP (*User Datagram Protocol*) i drugi. SOAP protokol smatra se žičanim protokolom (engl.*wire protocol*) jer ovisi o drugim protokolima. Ključni kriteriji koji se gledaju prilikom kreiranja žičanog protokola su kompaktnost, skalabilnost, interoperabilnost, učinkovitost protokola te prilagodljivost samog protokola na promjene.

SOAP poruka je dokument pisan u XML jeziku, a sastoji se od omotnice u kojoj se nalazi zaglavlje, koje sadržava detalje o poruci, tijelo u kojem se nalazi poruka koja se prenosi i koje može sadržavati poruke o greški. Blok za poruke o greški nije obavezan , no uvelike može olakšati pronalazak greške ukoliko se ona dogodila.

SOAP protokol se koristi za komunikaciju, no omogućeno je proširivanje za dodavanje sigurnosnih komponenata. Protokol ne pamti prošla spajanja i komunikaciju što znači da se svaka aplikacija prilikom slanja zahtjeva mora predstaviti drugim aplikacijma. [1] Specifičan je po malom broju grešaka koje se javljaju u komunikaciji i uslugama *WS-Security* koja omogućuje uključivanje dodatne sigurnosti za poduzeća, *WS-Atomic-Transaction* koja omogućuje isporuku valjanih podataka čak i u slučaju pojave greške i *WS-ReliableMessaging* koja osigurava određenu razinu pouzdanosti i sigurnosti u komunikaciji.

Promatrajući održavanje, lakše je održavati klijentsku, nego serversku stranu s obzirom da koristi WSDL datoteku u kojoj je opisano sučelje servisa. Komunikacija SOAP protkolom sastoji se od čvora pošiljatelja koji šalje poruku koja je trenutno kod njega, čvora primatelja koji prima poruku koja mu je upućena, putanje koja je zadana za određenu poruku, čvora inicijalnog pošiljatelja koji je kreirao poruku i poslao ju u komunikacijski kanal prema čvoru konačnog primatelja koji obrađuje pristiglu poruku te posrednika koji prilikom posjedovanja poruke obrađuju zaglavlja koja su namjenjena za njih.

Postoji dva načina komunikacije SOAP protokolom. Jednosmjerna komunikacija nema povratnog odgovora, cilj je ovakve komunikacije razašiljanje poruke i najčešće se koristi preko UDP protokola koji također ne očekuje povratnu potvrdu o primitku poruke. Dvosmjerna

komunikacija naziva se još *request/response* komunikacija u kojoj web servis razmjenjuje poruke odnosno podatke s drugim web servisom ili korisnikom.

3.1. SOAP omotnica

Omotnica SOAP protokola napisana je XML jezikom koji se koristi u većini web aplikacija. Vrlo jednostavan koncept nalik na HTML programski jezik, sastoji se od oznaka koje su ključne riječi u SOAP poruci.

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope
    xmlns:env="http://www.w3.org/2003/05/soap-envelope">
        <env:Header>
        <!-- mjesto za header . . . -->
        </env:Header>
        <env:Body>
        <!-- payload ili Fault . . . -->
    </env:Body>
</env:Envelope>
```

Slika 3.1.1 Prikaz SOAP omotnice

Omotnica sadrži deklaraciju imenika sa SOAP elementima i pravilima po kojima će se poruka obrađivati što je prikazano u trećem redu (Sl. 3.1.1), također sadrži zaglavlje koje sadrži upute gdje treba poslati SOAP dokument te dodatne komponente za sigurnost, kvalitetu usluge i druge.

Slika 3.1.2. Grafički prikaz SOAP omotnice

Tijelo je glavni dio omotnice i ono nosi korisne informacije odnosno poruku i poruke o pogrešci. Grafičkim prikazom sa slike 3.1.2. može se dobiti najopćenitiji uvid u osnovni izgled i raspored ključnih dijelova SOAP omotnice. Također se vidi hijerarhijsko slaganje oznaka prema prioritetima obrades koje je prikazano na slici 3.1.1.

3.1.1. Zaglavlje

Dio omotnice koji nije nužan za samu omotnicu no može sadržavati bitne podatke o načinu slanja i primanja, obrade, sigurnosne komponente, podatke o transakcijama, infromacije o kvaliteti usluge i drugima bez da se narušavaju specifikacije protokola. Pravilo je da zaglavlje uvijek dolazi prije oznake tijela SOAP poruke i uvijek mora biti prva podređena oznaka oznaci omotnice. Broj podređenih oznaka unutar zaglavlja nije ograničen, no velik broj oznaka može usporiti obradu SOAP XML dokumenta. Svaka oznaka unutar zaglavlja tvori zaglavni blok (engl. *header block*) koji je zadužen za sigurnost, transakcije i ostale ranije nabrojane mogućnosti i proširenja. [2] U zaglavnom bloku se može naći atribut *mustUnderstand* koja taj zaglavni blok označava kao bitan dio zaglavlja koji se ne smije ignorirati i uvijek mora biti obrađen prilikom

obrade SOAP poruke. [1] Ukoliko se ne obradi zaglavni blok s *mustUndestand* ključnom riječi cijela poruka se neće obraditi i javit će se greška.

3.1.2. Tijelo

Glavnina SOAP poruke nalazi se u tijelu poruke, koje je obavezno svakoj omotnici. Kao i zaglavlje, tijelo uvijek mora biti podređeno oznaci omotnice. Tijelo najčešće sadržava poruke o pogrešci ili podatke kojima se koriste aplikacije, no može se dogoditi i da tijelo ne sadrži ništa od navedenog. U tijelu se pozivaju metode pisane u određenom programskom jeziku, istim metodama se pridružuju vrijednosti za ispravno izvršavanje metoda. Također se u tijelu može naći blok za pogreške kojima se korisnika ili aplikaciju obavještava o pogreški i mjestu gdje se dogodila.

3.1.2.1. Blok za pogreške

[5] Poruke o pogrešci koje se mogu naći unutar tijela SOAP poruke unutar bloka za pogreške, a služe kako bi javile o pojavi greške u prijenosu ili pojavi greške prilikom čitanja i obrade poruke. Svaka poruka o pogreški sadrži pripadni kod pogreške definiran u specifikacijama SOAP protokola i razlog zbog kojeg je do pogreške došlo.

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
              xmlns:m="http://www.example.org/timeouts"
              xmlns:xml="http://www.w3.org/XML/1998/namespace">
 <env:Body>
  <env:Fault>
   <env:Code>
    <env:Value>env:Sender</env:Value>
    <env:Subcode>
     <env:Value>m:MessageTimeout</env:Value>
    </env:Subcode>
   </env:Code>
   <env:Reason>
    <env:Text xml:lang="en">Sender Timeout</env:Text>
   </env:Reason>
   <env:Detail>
    <m:MaxTime>P5M</m:MaxTime>
   </env:Detail>
  </env:Fault>
 </env:Body>
</env:Envelope>
```

Slika 3.1.2.1. Prikaz detalja o pogrešci

Moguće je i zatražiti detaljnije informacije o pogrešci ključnom rječi <soap-*env:Detail*>. Kao što je prikazano na primjeru detalj o pogrešci *Message Timeout* daje rezultat *<m:MaxTime>P5M</m:MaxTime>*.

Ime pogreške	Značenje
VersionMismatch	Zadani imenik nije ispravan i ne odgovara SOAP Envelope elementu (pr. poruka sadrži SOAP *upgrade* oznaku kojom koji navodi podršku za SOAP v1.1 i SOAP v1.2, ali sa postavkama za SOAP v1.2)
MustUnderstand	Dijete SOAP Zaglavlja sa MustUnderstand atributom postavljeno na vrijednost "1", nije uspješno shvaćena
Sender	Poruka je neispravno formirana ili sadrži neispravne podatke ili komunikacija sa serverom nije uspješno uspostavljena
Receiver	Dolazi do problema sa serverom pa se zbog toga poruka ne može ispravno precesirati
DataEncodingUnknown	Kodiranje podataka je nepoznato i nepodržano

Tablica 3.1.2.1. Prikaz detalja o pogrešci

U tablici su prikazani najčešći kodovi pogreške u SOAP 1.2 verziji protokola, dok se SOAP verzija 1.1 razlikuje u nekoliko kodova, naime *Sender*, *Receiver* i *DataEncodingUnknown* kodovi pogreške zamijenili su *Client* i *Server* kodove pogreške. Kod pogreške *Client* značio je isto kao i kod pogreške *Sender* u SOAP v1.2, a kod pogreške *Server* značio je isto kao i kod pogreške *Receiver* u SOAP v1.2.

3.2. XML

Jezik kojim su pisane SOAP poruke naziva se XML (engl. *Extensible markup Language*) što je skraćenica od *Proširivi Označni Jezik*. Upotrebljava se kao opisni jezik u web aplikacijama i web servisima za označavanje podataka. Vrlo sličan HTML jeziku sa svojim tagovima jednostavan je za čitati i naučiti. Izravno je primjenjiv na internetu te ga svi sustavi i sva računala vrlo brzo i efikasno procesiraju. Može povezati velike količine podataka u jedan dokument. Strukuriran je jezik zbog načina na koji je pisan, podatci se dijele u blokove. Služi za stvaranje formata informacija koje se prenose mrežnom komunikacijom.

U zaglavlju XML dokumenta se nalaze informacije o verziji XML jezika, pravilima o obradi teksta kao što je vidljivo u prvom retku (Sl. 3.1.1), dok u tijelu dokumenta stoji korisni dio poruke koja se prenosi. Svaki dokument ima pripadnu korijensku oznaku, u slučaju sa slike 3.1.1. to je oznaka *<env:Envelope>*, što govori kako se u ovom XML dokumentu prenosi SOAP omotnica s nekom porukom.

Prednost XML-a je u mogućnosti da korisnik sam definira svoje oznake i pri tome se mora pridržavati sintakse, u suprotnom poruka neće biti obrađena. Nedostatak mu je brzina koja opada u ovisnosti o veličini poruke, što je veća poruka, bit će potrebno više vremena za njenu obradu.

3.3. Oznake

Svaki redak pisan u XML dokumentu ili u SOAP omotnici sadrži određenu oznaku koja govori pregledniku i protokolima što se u tom retku nalazi kako bi znali procesirati istu poruku. Koriste se za kreiranje strukture unutar dokumenta. Neke oznake će imati roditeljske oznake (engl. *parent*), dok će druge imati oznaku djeteta (engl. *child*). Svaki XML dokument mora imati barem jednu glavnu roditeljsku oznaku koja sadrži podatke i uputu o obradi trenutnog dokumenta te verziji XML jezika u kojoj je dokument pisan. To je oznaka *<?xml version="1.0" encoding="UTF-8"?>*.

Za SOAP protokol i poruke koje se njime šalju ključne oznake su *<soap-env:Envelope>* koja govori da se radi o početku omotnice, zatim *<soap-ev:Header>* koja govori da se radi o zaglavlju koje sadrži bitne informacije o samom dokumentu i o načinu procesiranja istoga, potom *<soap-env:Body>* oznaka koja govori da se radi o tijelu SOAP poruke u kojoj je najčešće zahtjev upućen serveru ili aplikaciji ili je odogovor istih. Oznaka *<soap-env:Fault>* služi za obradu grešaka nastalih u komunikaciji putem SOAP poruka. Unutar bloka za pogreške nalaze se <soap-env:Code> i <soap-engl:Reason> oznake koje sadrže informacije o kodu pogreške i razlogu zbog kojeg je do iste došlo.

3.4. WSDL

Jezik kojim se opisuje ponašanje i radnje web servisa prema SOAP poruci i obratno naziva se WSDL (engl. *Web Service Description Language*). Njime se postavljaju apstraktna pravila prema kojima svaki web servis obrađuje pristigle poruke i šalje odgovore na iste. Također su opisana i sučelja kojima korisnik upravlja i djeluje nad podatcima.

Da bi se bolje shvatila uloga WSDL jezika potrebno je shvatiti što su web servisi. To su servisi odnosno usluge koje računala pružaju korisniku kako bi mogao komunicirati s drugima u internetskoj mreži na objektno orijentirani način te upravljati podatcima u bazi uz pomoć definiranog sučelja. Drugim riječima, pružaju mogućnost prikaza sučelja koje korisniku služe za interakciju sa aplikacijama, bazom i drugim korisnicima. Te interakcije kontrolirane su po pravilima zapisanim u WSDL datoteci, a neka od pravila reguliraju način na koji će zahtjev biti upućen web servisu, parametre koje web servis očekuje i s kojima će obavljati operacije.

```
<definitions>
    <types>
    <!--definicije tipova podataka...-->
    </types>

    <message name="GetUserRequest">
    <!--definicija poruke - zahtjeva...-->
        <part name="term" type="xs:string"/>
    </message>

    <message name="GetUserResponse">
    <!--definicija poruke - odgovora...-->
        <part name="value" type="xs:string"/>
    </message>

    <portType name="glossaryUsers">
    <!--definicija operacija koje se izvode po primitku zahtjeva...-->
        <operation name="GetUser">
            <input message="GetUserRequest"/>
            <output message="GetUserResponse"/>
        </operation>
    </portType>

    <binding type="glossaryUsers" name="bindingToSOAP">
    <!--informacije o protokolu i specifikacija formata podataka...-->
        <soap:binding style="document"
        transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation>
            <soap:operation soapAction="http://example.com/GetUser"/>
            <input><soap:body use="literal"/></input>
            <output><soap:body use="literal"/></output>
        </operation>
    </binding>
</definitions>
```

Slika 3.4.1. Prikaz spajanja WSDL dokumenta sa SOAP protokolom

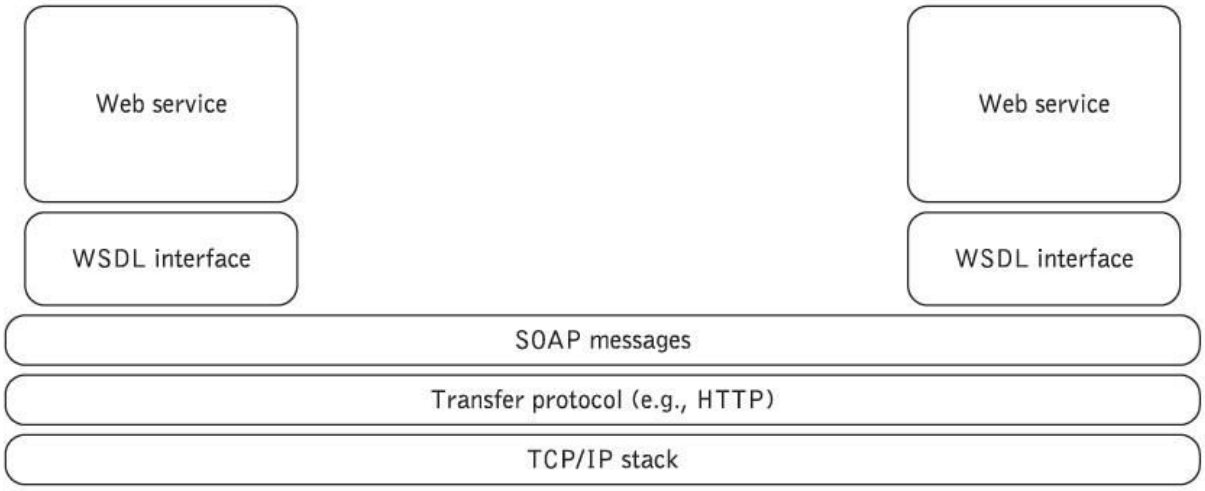
WSDL jezik je također jako sličan HTML-u po svojim oznakama, te vrlo razumljiv i jednostavan za pisanje. [6] Glavna, korijenska oznaka, *<definitions>* označava početak WSDL dokumenta u kojem su zatim opisana pravila oznakama *<types>* za definiciju tipova podataka u komunikaciji, zatim *<message>* za definiciju poruka u komunikaciji sa atributom *type* kojim se označava tip poruke koju treba obraditi, zatim *<portType>* za blok operacija. U tom bloku operacija nalaze se ulazna i izlazna poruka te naposlijetku *<binding>* za informacije o protokolu i

obliku podataka koji se šalje web servisu. Oznakom <*binding*> vrši se spajanje wsdl dokumenta s određenim protokolom namijenjenim za slanje poruka. Također sadrži ulaznu i izlaznu poruku, te SOAP akciju kojom usmjeravamo zahtjev na određenu metodu, u ovom slučaju *GetUser*. (Sl.3.4.1). Ključnom riječi *use* daje se do znanja da će tijelo ulazne i izlazne poruke biti obrađeno prema pravilima zadanim referencom u ključnoj rječi *transport*.

Moguće je poruke obrađivati na temelju referenci koje poruka nosi što se označava ključnom rječi *literal*, a svaka strana mora poznavati shemu i pravila za prijevod poruke. Također je poruke moguće obrađivati na temelju definiranih pravila i tipova podataka kojima će poruka biti obrađena, u ovom slučaju ta pravila se prenose zajedno s porukom.

4. PRINCIP RADA I PRIMJENA

SOAP protokol temelji se na razmjeni poruka između web servisa međusobno i web servisa i korisnika. Detalji o polazištu, odredištu, autoru te pravima pristupa nalaze se u zaglavlju svake omotnice.



Slika 4.1. Osnovni princip rada SOAP protokola

Svaka SOAP poruka koja je poslana aplikacijom, servisom ili serverom, koji koristi određeni web servis, prenosi se nekim transportnim protokolom. Taj protokol je u slučaju sa slike HTTP preko kojeg se šalje zahtjev serveru. Web servis po primitku zahtjeva otvara WSDL dokument u kojem su upute za obradu tog istog zahtjeva, zatim web servis procesira SOAP poruku i prikazuje ju korisniku uz pomoć sučelja ili kao tekstualni zapis. Nakon što je web servis primio poruku sa zahtjevom i obradio tu poruku rezultat obrade poslan je aplikaciji koja potom odlučuje kakav odgovor i koliko odgovora šalje nazad web servisu na obradu i zatim slanje korisniku ili web servisu koji je prvotno poslao zahtjev (Sl.4.1.)

4.1. Primjena

SOAP protokol primjenjuje se u web servisima namjenjenim kompatibilnosti sa što većim brojem transportnih protokola, što osigurava velike raspone korisnika, kako ljudi tako i aplikacija. Također se koristi zbog mogućnosti upravljanja greškama na način da ukoliko se

pojavi greška prilikom zahtjeva, odgovor na zahtjev će sadržavati kod pogreške (engl. *Error Code*) uz pomoću kojeg se problem može otkloniti. Za upravljanje greškama nastalim u komunikaciji SOAP protokolom mogu se kreirati liste sa kodovima pogreške koje korisniku daju detaljniji uvid zašto je došlo do pogreške te koje su mu preostale mogućnosti. [1] Najčešće se koristi u velikim vladinim sustavima i velikim poduzećima, sustavima za naplatu te bankarskim sustavima. SOAP protokol nije najkorišteniji protokol zbog većih zahtjeva u implementaciji i razumjevanja u odnosu na alternativu.

Najpoznatija primjena SOAP protokola danas se može naći u *Microsoftovim* proizvodima poput *Report Server Web service* koji omogućava izradu proizvoljnih alata za izvještaje, *Bing Maps* koristi SOAP api za pretraživanje karte svijeta, određivanje lokacije i drugih. Windows Web Services API je također zasnovan na SOAP protokolu a koristi se za aplikacije koje zahtjevaju malo vrijeme pokretanja, okruženja s ograničenom memorijom te aplikacije ne ovise o mnogo faktora. Paypal je još jedan u nizu korisnika koji koristi SOAP protokol zbog sigurnosnih aspekata koje pruža.

4.2. Primjer zahtjeva i odgovora

Kako bi zahtjev bio uspješno kreiran potrebno je u željenom programskom jeziku kreirati potrebne metode koje će biti pozivane unutar zahtjeva te je potrebno poznavati koje tipove podataka ta metoda može obraditi, također vrlo je bitno predati podatak u tom tipu kako ne bi došlo do greške.

```
<soap:Body xmlns:m="http://www.requestexample.com">
  <m:GetUserRequest>
    <m:UserID>123</m:UserID>
  </m:GetUserRequest>
</soap:Body>
```

Slika 4.2.1. Primjer jednostavnog zahtjeva za informacije o korisniku

U primjeru (Sl. 4.2.1.) je pokazan jednostavan SOAP zahtjev za informacije o korisniku u nekom sustavu. Sastoji se od imenika koji se nalazi na zadanoj lokaciji na internetu i govori na

koji način će se obrađivati podatci iz zahtjeva. Zatim sadrži i korisnikovu metodu pisanu u željenom programskom jeziku koja prima kao parametar korisnikovu identifikacijsku oznaku u brojčanom obliku tipa integer. Nakon primitka zahtjeva web servis po potrebi provjerava bazu podataka, provjerava je li došlo do greške, obrađuje zahtjev te priprema odgovor koji šalje korisniku ili aplikaciji koja je poslala zahtjev.

```
<soap:Body xmlns:m="http://www.requestexample.com">
  <m:GetUserResponse>
    <m:User>
      <m:Name>Bruno</m:Name>
      <m:Lastname>Balentovic</m:Lastname>
      <m:Age>22</m:Age>
      <m:Location>Osijek</m:Location>
      <m:Faculty>Faculty of Electrical Engineering,
      Computer Science and Information Technology Osijek</m:Faculty>
    </m:User>
  </m:GetUserResponse>
</soap:Body>
```

Slika 4.2.2. Primjer jednostavnog odgovora na zahtjev za informaciju o korisniku

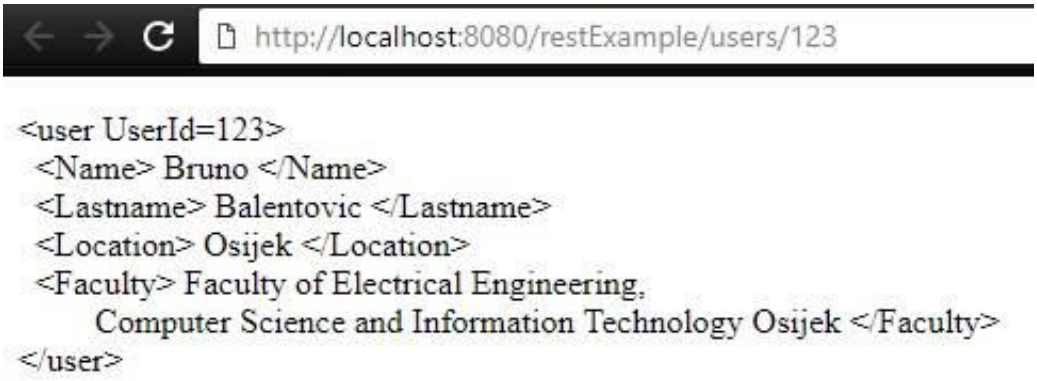
Svaki odgovor sadrži imenik identičan kao u zahtjevu jer se na taj način obrađuje željena interakcija s web servisom. U odgovoru se također nalazi korisnikova metoda, pisana u istom programskom jeziku kao ona iz zahtjeva, koja u ovoj situaciji obrađuje pristigle podatke iz zahtjeva te na temelju njih vraća svoj rezultat. Kako se u primjeru (Sl. 4.1.2.) radi o podatcima koji su najčešće spremljeni u bazu, metoda kontaktira bazu te na temelju pristigle vrijednosti *UserID* varijable vraća željene podatke.

Važno je napomenuti da SOAP odgovor ne može vratiti više informacija nego što je zatraženo, isto tako ne može se tražiti informacije koje ne postoje ili za koje korisnik ili aplikacija nemaju pravo pristupa.

5. ALTERNATIVNI PROTOKOL REST

Representational State Transfer koristi se kao alternativa SOAP protokolu zbog svoje jednostavnosti, brzine obrade zahtjeva, pouzdanosti, malog broja linija koda te mogućnosti za proširenje. [3] REST ne predstavlja protokol kao SOAP, već arhitekturu kojom je sve što REST koristi predstavljeno kao resurs, a svaki resurs označen prikladnim URI-jem (Uniform Resource Identifier). Skalabilnost je jedan od ključnih faktora zbog kojeg većina web servisa koristi upravo ovaj protokol, njome se omogućava rad nad velikim brojem interakcija s korisnicima, što je u današnje vrijeme jedan od preduvjeta za funkcioniranje web servisa na globalnoj razini.

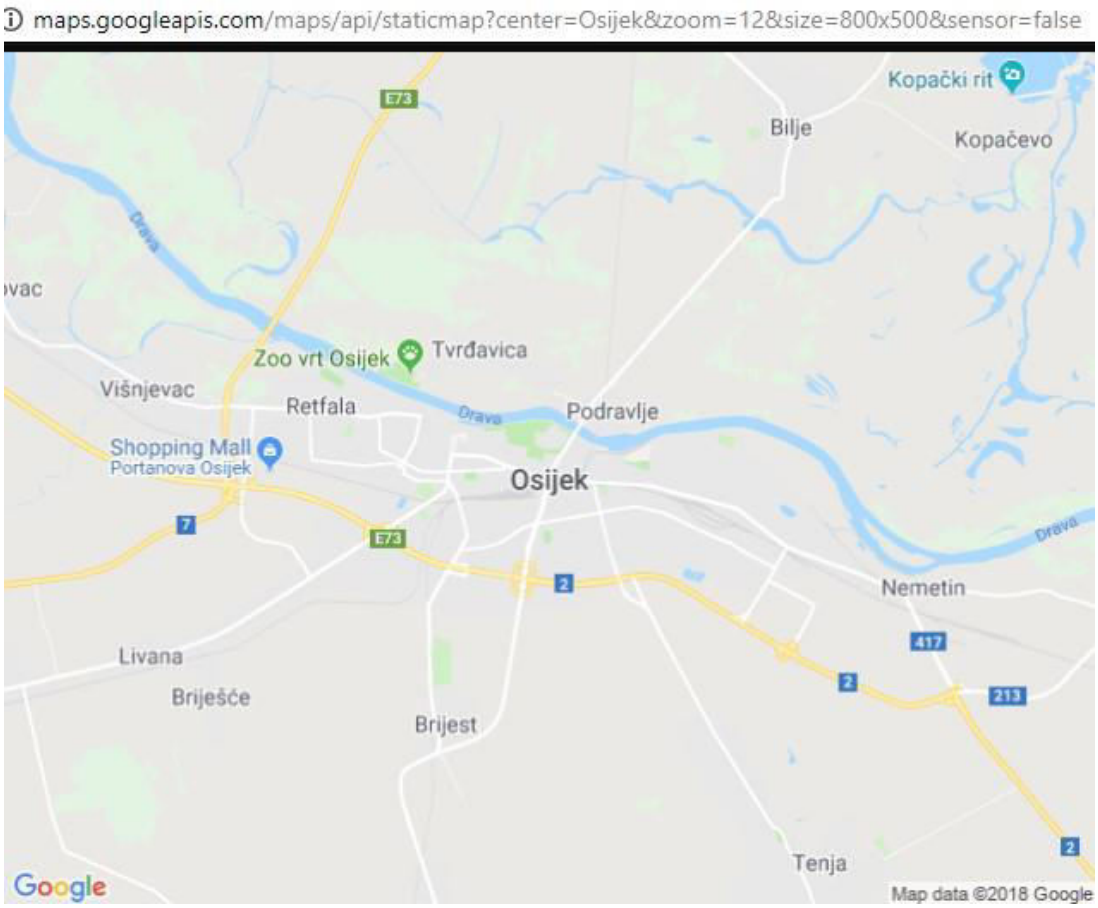
Prednosti u odnosu na SOAP protokol su brže reakcije na zahtjev, manja cijena implementacije, jednostavnije održavanje serverske strane komunikacije te manje korištenje memorije. [7] Najčešće se koristi u komunikaciji klijent–server za jednostavnu podatkovnu razmjenu tzv. od točke do točke komunikaciju (engl. *Point-to-Point*). REST je specifičan po slojevitom prikazu podataka i kretnjama kroz aplikaciju kako bi se došlo do podataka, to se postiže korištenjem URL-a (Uniform Resource Locator) . Korisnik se nastoji postepeno kretati kroz aplikaciju tako što odabire linkove koji ga vode do željenog podatka na kojemu uz pomoć CRUD (engl. *CREATE*, *READ*, *UPDATE*, *DELETE*) metoda obavlja radnje.



Slika 5.1. Primjer jednostavnog REST zahtjeva i odgovora pisanog u XML-u

U navigacijskoj traci postavljen je zahtjev koji se sastoji od putanje koja vodi do tablice *users* iz baze podataka te zahtjeva ispis podataka o korisniku kojem je pridružen navedeni id. Kao odgovor na zahtjev dobiva se, u ovom slučaju, XML ispis podataka iz baze. Vrlo čitljiv i korisniku razumljiv prikaz odgovora. Glavna svojstva koja REST razlikuju od SOAP-a su veće

brzine kojima REST obrađuje podatke, manji broj linija koda, lakša implementacija, lakše i brže učenje, REST može koristiti više načina za odgovor kao što su XML, JSON i drugi.



Slika 5.2. REST primjer na Google Maps

Neke od poznatijih uporaba REST načina rada sa web servisima su Google Maps, Google Translate, Google Earth, Twitter, Yahoo, Amazon, Facebook i drugi . U primjeru (Sl. 5.2) se vidi kako korisnik zadaje određene parametre kao što su veličina slike rezultata, zoom slike, postojanost senzora te što će se nalaziti u centru slike rezultata.

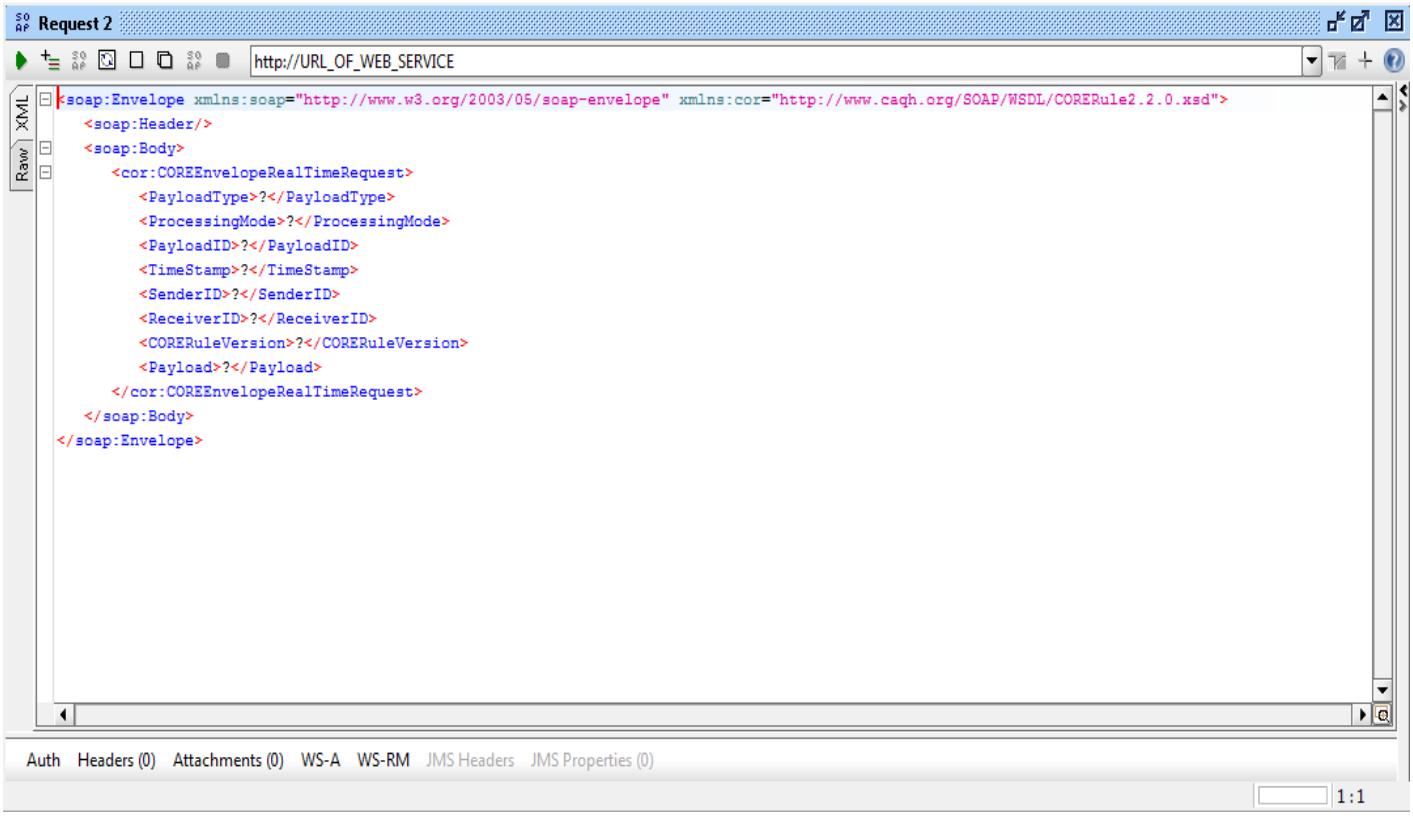
Kategorija	SOAP	REST
Puno ime	Simple Object Access Protocol	Representational State Transfer
Starost	1998.	1994.
Potreban broj linija koda za implementaciju	Manji broj linija koda za implementaciju	Veći broj linija koda za implementaciju
Cijena održavanja i implementacije	Veća	Manja
Brzina izvršavanja	Manja	Veća
Količina pogrešaka	Manja	Veća
Pouzdanost	Veća	Manja
Performanse	Manje	Veće
Neovisnost o protokolu	Neovisan o protokolu	Ovisi o HTTP
Protokol preko kojega se može koristiti	HTTP, TCP, UDP, SMTP	HTTP
Jezik kojim su podaci oblikovani	XML	XML, JSON
Specifične karakteristike	-Siguran prijenos podataka -Digitalni potpisi -Sigurna razmjena povjerljivih poruka -Tajnost -Kredibilitet -Konzistencija -Skalabilnost -Interoperabilnost -Prilagodljivost -Protokol	-Velike brzine obrade podataka -Laka implementacija -Lako održavanje -Korištenje URI-ja za označavanje podataka -Vrlo čitljiv -Lako ispravljanje poruka -Jednostavan za naučiti -Sve veća upotreba -Stil kojim je arhitektura opisana

Tablica 5.1. Usporedba SOAP i REST

Iako REST nije protokol kao SOAP, vrlo često se dovodi pitanje „Koji je bolji?“, a na to pitanje se može odgovoriti ukoliko se pogleda Tablica 5.1 u kojoj se nalaze osnovne karakteristike te osnovne usporedbe navedenih pojmova.

5.1.SoapUI aplikacija

[8] SoapUI je aplikacija otvorenog koda čiji je autor tvrtka SmartBear koja se bavi proizvodnjom aplikacijskih programskih sučelja (eng. API), kreiranjem softvera, kreiranjem alata za nadzor performansi te testiranjem softvera. U SoapUI aplikaciji mogu se testirati i SOAP i REST rješenja. U aplikaciji korisnik koji želi testirati svoje rješenje piše testove na temelju ranije spomenute WSDL datoteku koju mora priložiti u aplikaciju. Nakon toga korisniku se prikazuje zahtjev koji nakon pokretanja testa vraća odgovor. U slučaju SOAP zahtjeva aplikacija će kao odgovor vratiti SOAP omotnicu sa prikladnim odgovorom, u slučaju REST zahtjeva temeljenog na GET metodi aplikacija kao odgovor vraća XML zapis sa prikladnim odgovorom ili mrežnu adresu.



Slika 5.1.1. Primjer SOAP zahtjeva u SoapUI aplikaciji

Slika 5.1.1. prikazuje jednostavan primjer SOAP zahtjeva u SoapUI aplikaciji, koji se generira nakon prilaganja WSDL datoteke u projekt. SoapUI aplikacija će automatski staviti znak "?" metodi koja očekuje nekakav unos od korisnika ili aplikacije koja rukuje web servisom. Na

mjesto znaka "?" potrebno je staviti odgovarajući tip podatka koji određena metoda očekuje kako bi testiranje prošlo uspješno. Ukoliko se na traženo mjesto ne stavi traženi tip podatka ili ispravan podatak odgovor neće biti ispravan, ili će ga blok za pogreške, ukoliko je implementiran, obraditi i ispisati pogrešku. Aplikacija SoapUI nije nužna za rad ni REST-a ni SOAP-a, nju se koristi isključivo za testiranje i provjeru ispravnosti web servisa kreiranih na nekom od dva načina.

6. ZAKLJUČAK

U vremenu kada se za komunikaciju putem interneta nastoji odvojiti što manje vremena te nastoji što kraće čekati željeni odgovor, bilo je potrebno kreirati protokol koji će omogućiti tekstualnu razmjenu poruka velikom brzinom, što većem broju korisnika istovremeno. Zbog te potrebe nastao je SOAP protokol koji definira način uklapanja poruka oblikovanih u XML jeziku u neki od dostupnih transportnih protokola. S obzirom na to da u nekim situacijama određeni transportni protokol nije dostupan, a SOAP poruka mora biti poslana, odabire se drugi najpogodniji transportni protokol za prijenos poruke.

Ukoliko se SOAP promatra sa aplikacijske strane, mogu se uvidjeti mnoge mane i nedostaci, no razlog opstanka SOAP protokola je moguća proširivost dodacima za sigurnost, konzistenciju, izolaciju i izdržjivost podataka kao i kompaktnost, skalabilnost, prilagodljivost protokola na promjene, interoperabilnost, učinkovitost protokola, neovisnost o operacijskom sustavu, protokolu i programskom jeziku u kojem je poruka pisana.

Za donošenje odluke koji način komunikacije se želi implementirati u neki sustav za razmjenu poruka potrebno je promotriti aspekte koje taj sustav pokriva i slučajeve u kojima se može naći. Ovisno o tome je li namjena siguran prijenos podataka, tajnost, kredibilitet, digitalno potpisivanje i razmjena privatnih povjerljivih podataka odabrat ćemo SOAP protokol. U slučaju kada nam oblik izlaznog podatka nije bitan, već nam je bitna informacija, kada je bitna brzina, mrežna propustnost u mobilnim aplikacijama odabrat ćemo REST arhitekturu. Većina današnjih programera odlučuje se upravo za REST arhitekturu zbog jednostavnosti programiranja i lakše razumljivosti, ali ako je potrebna povratna informacija o ispravnosti i uspješnosti isporuke poruke ili uspješnosti izvršenja naredbe unutar SOAP poruke odabrati će SOAP. Također SOAP će biti odabran ukoliko je potrebna brza izmjena kratkih poruka koje nose podatke o autentikaciji ili autorizaciji korisnika telekomunikacijskih usluga. Ključna karakteristika SOAP protokola je mogućnost pristupa aplikacijama bez smetnji vatrozida. Iako postoje mnoge, novije i popularnije alternative SOAP protokolu, njegova budućnost je i dalje svijetla.

7. LITERATURA

[1] *Web Services: Principles and Technology -Michael P. Papazoglou*, *Tilburg University Netherland* 2008.

[2] *W3C SOAP v1.2 - https://www.w3.org/TR/soap12/*

[3] *SOAP, Wikipedia - https://en.wikipedia.org/wiki/SOAP*

[4] *Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics, Juris Tihomirovs i Janis Grabis*, 2016

[5] *IETF: SOAP- https://tools.ietf.org/html/draft-box-http-soap-00*

[6] *W3C WSDL v2.0 - https://www.w3.org/TR/2007/REC-wsdl20-20070626/*

[7] *Pametna komunikacija na Internetu preko REST protokola,Davor Lozić i Alen Šimec*, 2014.

[8] *SoapUI - https://www.soapui.org/soapui-projects/soapui-projects.html*

Popis korištenih oznaka/kratica:

SOAP – Simple Object Access Protocol

HTTP – Hyper Text Transfer Protocol

SMTP – Simple Mail Transfer Protocol

TCP – Transport Control Protocol

UDP – User datagram Protocol

XML – Extensible Markup Language

HTML – Hyper Text Markup Language

WSDL – Web Service Description Language

UDDI – Universal Description Discovery and Integration

W3C – World Wide Web Consortium

SAŽETAK

Rad se bavi temom komunikacijskog protokola koji može biti primjenjen na svim transportnim protokolima današnjice. Prikazuje povijest SOAP protokola te navodi razloge zbog kojih je pogodan za uporabu. Na jednostavnim primjerima pokazuje kako ga koristiti. SOAP protokol se najčešće koristi u velikim sustavima u kojima je sigurnost prioritet, služi za obavljanje raznih transakcija u bankovnim sustavima, naplatnim sustavima i vladinim sustavima i velikim poduzećima. Detaljno su opisani građa SOAP omotnice, najčešće korištene XML oznake unutar jedne omotnice kao i jezik WSDL koji služi za opisivanje web servisa. Objašnjen je pojam REST, koji je naveden kao alternativno rješenje mrežne komunikacije. Također su navedene prednosti i nedostaci u odnosu na SOAP. Usprkos nedostatcima u odnosu na REST, razlog opstanka i daljnjeg korištenja SOAP-a kao protokola i načina rada sa web servisima leži u većoj sigurnosti, proširivosti sa raznim ekstenzijama i mogućnosti funkcioniranja na raznim transportnim protokolima.

Ključne riječi:

SOAP protokol, REST, XML, WSDL, web servisi, omotnica, zahtjev, odgovor

ABSTRACT

This paper deals with the theme of communication protocol that can be applied on most of transport protocols these days. Shows history of SOAP protocol and lists reasons why is it suitable for use. On simple examples shows how to use it. SOAP protocol is mostly used in enterprise systems where the main priority is security and data, used for transactions in bank systems, payment systems, government systems and big companies. The SOAP envelope structure, most frequently used XML tags in one envelope so as language WSDL for describing web services are described in detail. Explained the term REST, which is listed as alternative solution in web communication. As well as advantages and disadvantages are listed in regards to SOAP. Despite disadvantages compared to REST, reason for SOAP's existence and further usage as a protocol so as the way of working with webservices lies in bigger security, expandability and possibility of functioning on different transport protocols.

Keywords:

SOAP protocol, *REST, XML,WSDL, web services, envelope, request, response*

ŽIVOTOPIS

Bruno Balentović rođen je u Osijeku, 27. kolovoza 1996. godine. Osnovnu školu završava 2011. u Dardi, nakon čega upisuje I.gimnaziju u Osijeku. U Osijeku 2015. godine završava I.gimnaziju te upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Redoviti je student stručnog studija Informatike. Na fakultetu se prvi puta susreće sa programiranjem i pojmom mrežne komunikacije. Zainteresiran na području mrežne komunikacije i backend developmenta. Od pruženih znanja na fakultetu najviše uživa u web programiranju, i back-end i front-end. Sebe vidi kao back-end developera. Neki od programskih jezika kojima se služi i zna koristiti su SQL, Java, PHP, C++, C, HTML, CSS, JavaScript, XML. U potpunosti razumije engleski jezik.

Bruno Balentović