

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1524

**Praćenje korištenja sredstava usluga putem
posrednika**

Ivor Sučić

Zagreb, ožujak 2005.

Zadatak

Opisati standarde i tehnologije koje omogućavaju pružanje Web usluga putem računalne mreže Internet. Posebno proučiti WS-Resources tehnologiju, te njene mogućnosti adresiranja pojedinog sredstva unutar Web usluge. Proučiti postojeća rješenja praćenja i bilježenja korištenja Web usluga, a posebnu pažnju posvetiti praćenju Web usluga u MidArc posredniku. Unaprijediti način praćenja korištenja usluga MidArc posrednika dodavanjem mogućnosti praćenja usluga zasnovanih na WS-Resources tehnologiji. Izgraditi sustav praćenja korištenja u .Net tehnologiji. Opisati arhitekturu unaprijeđenog sustava praćenja korištenja usluga, navesti korištenu literaturu i primljenu pomoć.

Sadržaj

1. Uvod	1
2. Razvoj usluga na Internetu	3
2.1. Razvoj Web usluga	8
2.2. Razvoj Web Services usluga	9
2.2.1. HTTP	13
2.2.2. XML	15
2.2.3. SOAP	18
2.2.4. WSDL	22
2.3. Proširenja Web Services tehnologije	24
2.3.1. Grid usluge	26
2.3.2. WS-Resources	29
3. Praćenje korištenja usluga	32
3.1. Metode praćenja korištenja usluga	33
3.1.1. Metoda zasebnog prikupljanja i pretvorbe podataka	33
3.1.2. Metoda istovremenog prikupljanja i pretvorbe podataka	34
3.2. Područja primjene praćenja korištenja usluga	35
3.2.1. Praćenje korištenja usluga radi osiguravanja ispravnog rada	35
3.2.2. Praćenje korištenja usluga radi naplate	36
3.2.3. Praćenje korištenja usluga radi izrade korisničkih profila	37
4. Posrednički sustav MidArc	39
4.1. Funkcionalnosti MidArc posrednika	41
4.2. Način rada MidArc posrednika	43
4.3. Arhitektura MidArc posrednika	44
5. Organizacija i nadogradnja podsustava za praćenje korištenja usluga	46
5.1. Analiza metode praćenja korištenja usluga u sklopu MidArc posrednika ...	46
5.2. Arhitektura podsustava za praćenje korištenja usluga	49
5.3. Nadogradnja modula za registraciju usluga	50
5.3.1. Modul za zadavanje pravila praćenja korištenja usluga	52
5.3.2. Struktura predloška za praćenje korištenja usluge	54
5.3.3. Grafičko sučelje za zadavanje pravila praćenja korištenja usluga.....	57
5.4. Modul za bilježenje korištenja usluga	59
6. Zaključak	62
7. Literatura	64

1. Uvod

Prije uspostavljanja WWW-a (engl. *World Wide Web*), Internet su koristile akademske i vladine institucije. Web je nastao devedesetih godina prošloga stoljeća u znanstvenoj zajednici i za znanstvene potrebe. S vremenom, Web se razvio u okruženje putem kojega su širokoj populaciji postale dostupne informacije različitog sadržaja. Uz stalni porast broja korisnika proširuje se i ponuda i raznovrsnost sadržaja dostupnih putem Weba. Danas su tako, osim ponude informacija, na Webu razvijene i razne druge naprednije usluge koje nude elektroničko bankarstvo (engl. *e-banking*), *on-line* aukcije, Web dućani, elektronička birokracija (engl. *e-government*) i drugo.

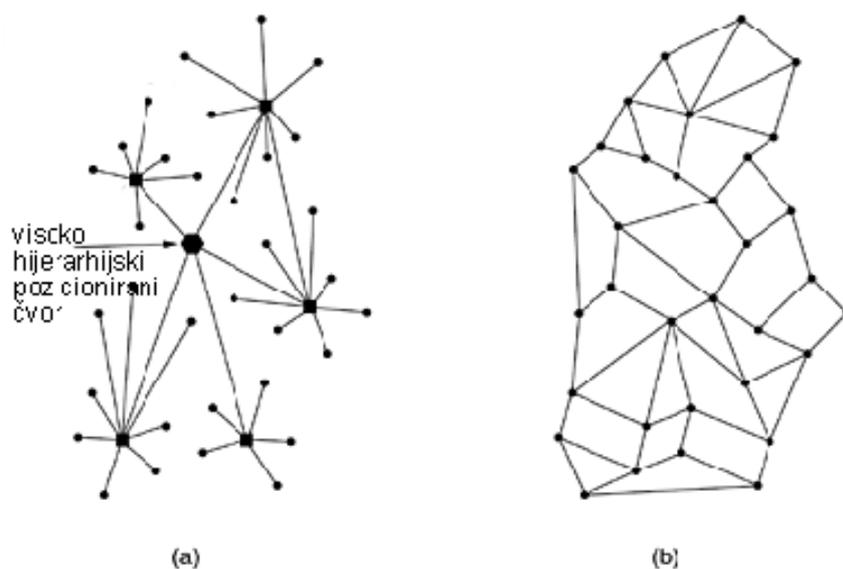
Osnovni dio napredne usluge je poslovna logika koja se izlaže i nudi korisnicima. Međutim, pored poslovne logike, napredne usluge obuhvaćaju i skup zajedničkih funkcionalnosti potrebnih za potporu izvođenja poslovne logike. Zajedničke funkcionalnosti obuhvaćaju registraciju i autentikaciju korisnika, autorizaciju korištenja usluga, praćenje korištenja usluga, sigurnost i drugo. Razvoj i ugradnja zajedničkih funkcionalnosti u svaku uslugu utječe na tehničku zahtjevnost, vrijeme i cijenu razvoja usluga. Kako zajedničke funkcionalnosti nisu u izravnoj vezi s poslovnom logikom usluge, moguće ih je izdvojiti iz pojedinačnih usluga i smjestiti u zaseban sustav – posrednički sustav. Posrednički sustav je središnji sustav koji posreduje u komunikaciji između korisnika i usluga. Umjesto da svaka usluga zasebno razvija skup zajedničkih funkcionalnosti, zajedničke funkcionalnosti razvijene su jednom, te se izvode na posredniku. Uvođenje posredničkog sustava pojednostavljuje i skraćuje razvoj usluga, te smanjuje troškove razvoja usluga. Osim toga, posrednik jamči za kvalitetu zajedničkih funkcionalnosti jer one više nisu sporedni dio usluge, već postaju glavni dio posredničkog sustava.

U ovom radu obrađena je funkcionalnost praćenja korištenja usluga, a posebna pažnja posvećena je praćenju korištenja sredstava usluga putem posrednika. U sklopu praktičnog dijela rada unaprijeđen je podsustav za praćenje korištenja usluga posredničkog sustava MidArc koji se razvija u suradnji Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu i tvrtke Ericsson Nikola Tesla. Korištenje usluga prati se za potrebe naplaćivanja, te za potrebe izrade korisničkih profila.

U drugom poglavlju rada opisan je razvoj usluga na Internetu. Dan je pregled tehnologija u kojima se usluge ostvaruju. Objasnjeni su razlozi koji su doveli do nastanka pojedinih tehnologija, te utjecaj pojedinih tehnologija na razvoj usluga. U trećem poglavlju prikazane su metode kojima se danas na Internetu prati korištenje usluga. Iznijet je pregled najčešćih razloga praćenja korištenja usluga, te je objašnjeno koje se metode pritom koriste. MidArc posrednički sustav opisan je u četvrtom poglavlju. Opisane su funkcionalnosti i arhitektura MidArc sustava. Peto poglavlje prikazuje podsustav za praćenje korištenja usluga izgrađen u sklopu praktičnog dijela ovoga rada. Opisana je metoda praćenja korištenja usluga koja je pritom korištena. Prikazana je arhitektura i programsko ostvarenje modula. Zaključak diplomskog rada iznesen je u šestom poglavlju.

2. Razvoj usluga na Internetu

Počeci Interneta sežu u šezdesete godine prošloga stoljeća. Tada se za komunikaciju koristila isključivo telefonska infrastruktura (engl. *Public Switched Telephone Network*). Budući da je telefonska infrastruktura organizirana hijerarhijski, s malom količinom redundancije, smatralo se da je previše osjetljiva za vojne potrebe. Prestankom rada samo jednog visoko hijerarhijski pozicioniranog čvora telefonska mreža se raspada u nekoliko manjih međusobno nepovezanih mreža (slika 1). Zbog toga je ministarstvo obrane SAD-a, zajedno s nekoliko američkih sveučilišta, pokrenulo projekt čiji je cilj bio razvoj pouzdanije komunikacijske infrastrukture. Pouzdanost nove komunikacijske infrastrukture zasnivala se na redundanciji. Između svaka dva komunikacijska čvora postojalo je više komunikacijskih puteva, čime se smanjila mogućnost raspada mreže u više nepovezanih manjih mreža.

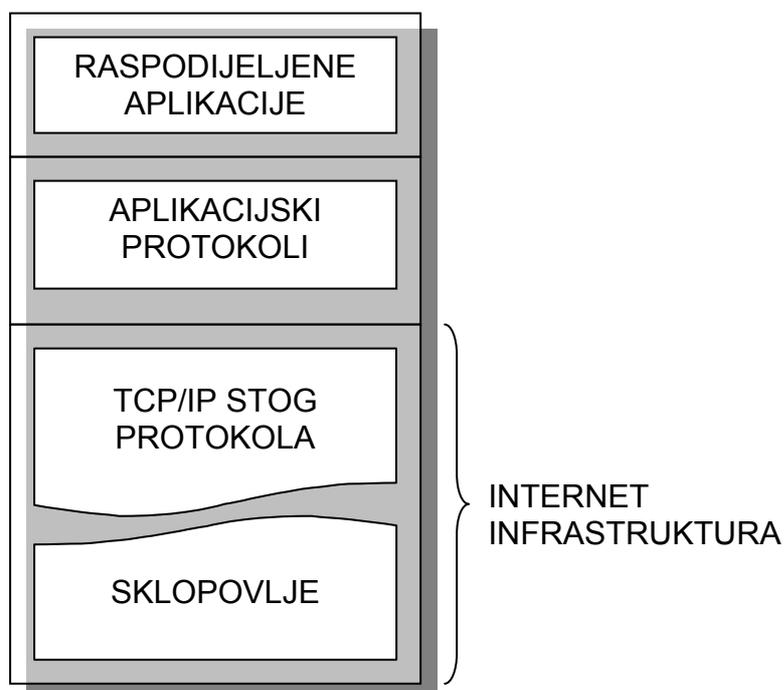


Slika 1. Telefonska (a) i Internet (b) komunikacijska infrastruktura.

Razvijena infrastruktura omogućila je znanstvenicima sa sveučilišta uključenih u projekt naprednu razmjenu podataka i suradnju na istraživačkim projektima. Kao rezultat, rastao je broj sveučilišta uključenih u projekt, te se javlja ideja da se sva američka sveučilišta povežu u jedinstvenu istraživačku mrežu. Međutim, dotad razvijeni komunikacijski protokoli pokazali su se nedostatnim za uspostavljanje komunikacije između velikog broja različitih lokalnih mreža na sveučilištima. Zbog

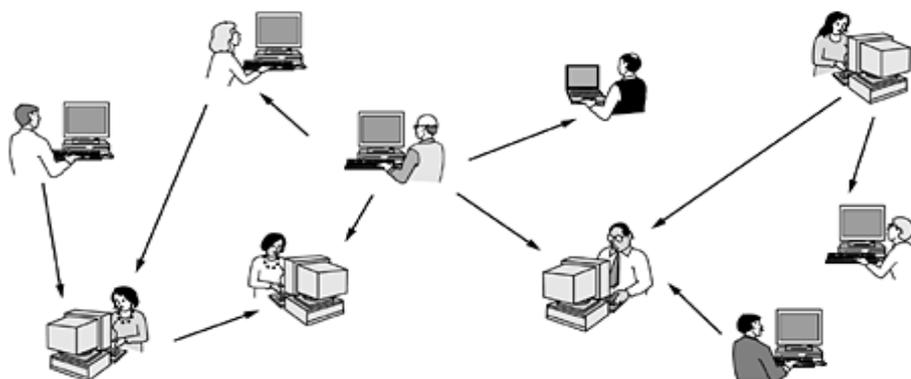
toga su pokrenuta istraživanja koja su rezultirala stvaranjem TCP/IP (engl. *Transmission Control Protocol, Internet Protocol*) protokola. TCP/IP protokol zasniva se na stogu koji je dizajniran kako bi omogućio komunikaciju između različitih računalnih mreža. Istraživačke mreže, slične onoj koja je povezivala američka sveučilišta, razvijale su se i u drugim državama i regijama. Sredinom osamdesetih godina prošlog stoljeća TCP/IP stog prihvaćen je kao službeni prijenosni protokol. Prihvatanje jedinstvenog prijenosnog protokola omogućilo je spajanje različitih istraživačkih mreža su u ono što danas nazivamo Internet.

Globalnu mrežu Internet može se ugrubo podijeliti na dva komunikacijska sloja [1], kao što je prikazano na slici 2. Povrh komunikacijskih slojeva smješten je sloj aplikacija koje komuniciraju koristeći pritom navedene niže slojeve. U donji sloj ubrajaju se standardi i protokoli od samog sklopovlja do stoga protokola TCP/IP. To je sloj Internet infrastrukture. Ona omogućuje uspostavljanje dvosmjerne komunikacije (engl. *full duplex*) između dva ravnopravna sudionika. Međutim, takva infrastruktura ne zadovoljava sve potrebe za uspješnu komunikaciju među aplikacijama. Razlika između komunikacijskih mogućnosti koje pruža infrastruktura i potreba za uspostavljanje uspješne komunikacije među aplikacijama premoštena je slojem aplikacijskih protokola.



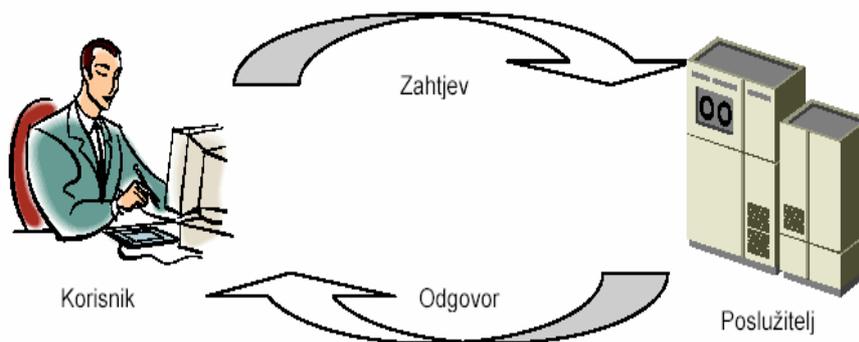
Slika 2. Podjela globalne mreže Internet na slojeve.

Korištenjem različitih aplikacijskih protokola moguće je ostvariti različite oblike komunikacije. Najkorišteniji oblici komunikacije danas na Internetu su komunikacija korisnik-poslužitelj (engl. *client-server*) i komunikacija ravnopravnih sudionika (engl. *peer-to-peer*). U komunikaciji među ravnopravnim sudionicima (slika 3) ne postoji podjela među sudionicima prema njihovoj ulozi u komunikaciji. Unatoč tome što se sudionici u komunikaciji mogu međusobno razlikovati, ravnopravni su u mogućnostima i funkcionalnostima. Za razliku od navedenog, u komunikaciji prema komunikacijskom modelu korisnik-poslužitelj (slika 4) sudionici se dijele se na korisnike (engl. *client*) i poslužitelje (engl. *server*). Uloga poslužitelja je da odgovaraju na zahtjeve korisnika, dok je uloga korisnika upućivanje zahtjeva.



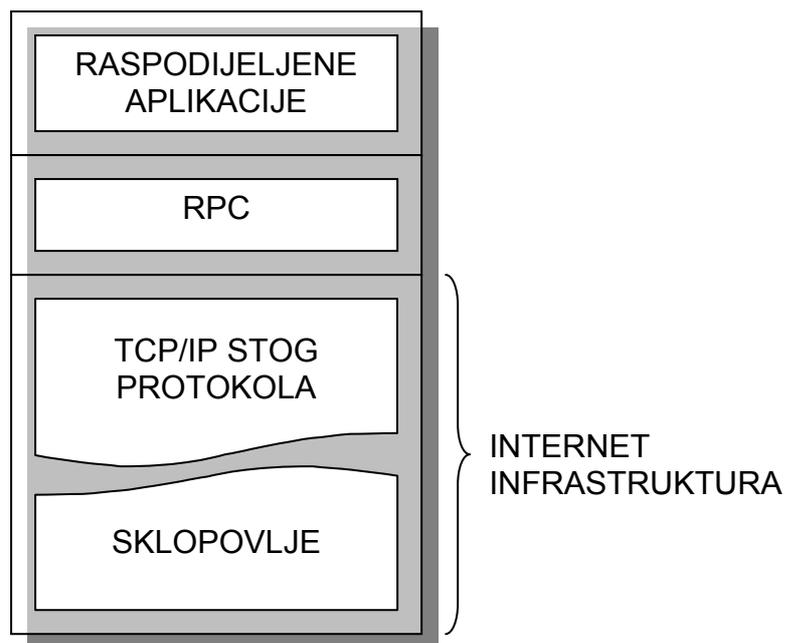
Slika 3. Komunikacija prema *peer-to-peer* komunikacijskom modelu.

Pojam usluge u računarstvu se prvi puta javlja u sklopu korisnik-poslužitelj komunikacijskog modela. Poslužitelji su uglavnom napravljeni tako da odgovaraju na manji broj različitih vrsta zahtjeva. Pritom obično postoji neka logička veza među njima. Skup logički povezanih vrsta operacija koje poslužitelj obavlja na zahtjev korisnika naziva se uslugom. Tako, na primjer, poslužitelj elektroničke pošte (engl. *mail server*) odgovara isključivo na zahtjeve vezane uz slanje i primanje elektroničke pošte, odnosno pruža uslugu slanja i primanja elektroničke pošte.



Slika 4. Komunikacija prema korisnik-poslužitelj komunikacijskom modelu.

Usluge možemo ugrubo podijeliti na dvije vrste – usluge orijentirane razmjeni podataka i usluge orijentirane izvođenju akcija. Usluge orijentirane razmjeni podataka odgovaraju na zahtjeve koji uključuju slanje i pristup udaljenim podacima. Usluge orijentirane izvođenju akcija odgovaraju na zahtjeve za izvođenjem udaljenog programskog koda. Za komunikaciju s uslugom izvođenja udaljenog programskog koda razvijeni su standardni protokoli aplikacijske razine, te pripadajući komunikacijski mehanizmi. Korištenje navedenih komunikacijskih mehanizama pojednostavljuje razvoj raspodijeljenih aplikacija koje koriste uslugu izvođenja udaljenog programskog koda. Komunikacijski mehanizmi brinu se za komunikaciju korisničkog i poslužiteljskog dijela raspodijeljene aplikacije. Sastoje se od dijela koji se izvodi na korisničkoj strani (engl. *client stub*), te dijela koji se izvodi na poslužitelju (engl. *server stub*). Programeri raspodijeljenih aplikacija koriste navedene mehanizme, te ne moraju brinuti za komunikaciju korisničkog i poslužiteljskog dijela aplikacije. Razvijeno je više različitih standarda za komunikaciju s uslugama izvođenja udaljenog programskog koda. Svim razvijenim standardima zajedničko je što s programerskog stanovišta pristup poslužitelju izgleda kao poziv procedure. Takav pristup poslužitelju naziva se poziv udaljene procedure. Protokol aplikacijske razine i mehanizmi za komunikaciju korisničkog i poslužiteljskog dijela raspodijeljene aplikacije zajedno čine komunikacijski sloj koji se označava kraticom RPC (engl. *Remote Procedure Call*). Smještaj RPC sloja na Internetu prikazan je na slici 5.



Slika 5. Smještaj RPC sloja na Internetu.

Do kraja 80-ih godina prošlog stoljeća svaka usluga orijentirana podacima imala je vlastiti aplikacijski protokol za komunikaciju korisnika i poslužitelja usluge. U tom razdoblju nastalo je mnoštvo različitih aplikacijskih protokola. Razvijeno je nekoliko protokola za komunikaciju s poslužiteljem elektroničke pošte, od kojih su najpoznatiji SMTP [2] (engl. *Simple Mail Transfer Protocol*) i POP3 [3] (engl. *Post Office Protocol Version 3*) protokoli. Zatim, razvijeni su protokoli za pristup datotečnom poslužitelju, kao na primjer FTP [4] (engl. *File Transfer Protocol*) i protokoli za udaljeni pristup računalu (npr. telnet protokol). Također, razvijeni su i protokoli za objavljivanje i čitanje novosti s odgovarajućeg poslužitelja (engl. *USENET news*), i drugi.

Princip prema kojem se pri razvoju nove usluge razvija i odgovarajući aplikacijski protokol za komunikaciju korisnika i poslužitelja usluge kočio je razvoj usluga. Problem je bio u tome što je za svaku novu uslugu prvo bilo potrebno distribuirati korisnički dio aplikacije koji razumije aplikacijski protokol usluge. Pojednostavljenje razvoja usluga donio je tek razvoj i usvajanje Web tehnologija. Korištenje Web tehnologija omogućilo je razvoj usluga kojima se pristupa pomoću jedinstvenog aplikacijskog protokola, te jedinstvenog korisničkog dijela aplikacije.

2.1. Razvoj Web usluga

World Wide Web, ili kraće samo – Web [1], čini skup tehnologija prihvaćenih kao aplikacijski protokol za komunikaciju. Web je nastao 1989. godine u CERN-u, Europskom centru za nuklearna istraživanja. U početku, Web je bio samo još jedna raspodijeljena aplikacija, s pripadajućim aplikacijskim protokolom, razvijena u skladu s korisnik-poslužitelj komunikacijskim modelom. Zamišljen je kao aplikacija za pristup informacijama organiziranim u povezane dokumente. Korisničku stranu aplikacije čini Web preglednik (engl. *Web browser*). Putem Web preglednika korisnik šalje poslužitelju zahtjeve za dokumentima, te ih po primitku prikazuje na zaslonu. Poslužitelj (engl. *Web server*) čeka na zahtjev korisnika, a po primitku zahtjeva kao odgovor šalje traženi dokument.

Web se zasniva na dva standarda – aplikacijskom protokolu HTTP (engl. *Hyper Text Transfer Protocol*), te jeziku HTML (engl. *Hyper Text Markup Language*). HTTP je protokol za prijenos zahtjeva i odgovora između Web preglednika i Web poslužitelja, dok se jezik HTML koristi za zapisivanje dokumenata. Dokumenti u HTML formatu sastoje se od teksta dokumenta isprepletenog uputama za formatiranje. HTML je omogućio povezivanje dokumenata na način da pojedine riječi u tekstu jednog dokumenta sadrže reference na druge dokumente. Takav tekst dobio je naziv *hyper text*, prema čemu je i jezik dobio svoje ime.

Razvoj novih tehnologija unaprijedio je mogućnosti Weba. Razvijene tehnologije mogu se podijeliti u dvije skupine – tehnologije poslužiteljske logike (engl. *server side scripts*) i tehnologije korisničke logike (engl. *client side scripts*). Tehnologije poslužiteljske logike omogućavaju da se na Web poslužitelju po primitku zahtjeva izvede proizvoljni programski odsječak kojim se dinamički stvara odgovor. Dinamički odgovor stvara se bez utjecaja korisnika koji ne može razlikovati je li dokument dinamički stvoren ili nije. Osim stvaranja samog dokumenta, koji se zatim šalje kao odgovor, poslužiteljska logika može izvesti i druge radnje. U praksi se najčešće koristi za dohvat postojećih ili upis novih podataka u bazu podataka. Najčešće korištene tehnologije poslužiteljske logike su ASP i ASP.NET (engl. *Active Server Pages*), PHP (engl. *Hypertext Preprocessor*), JSP (engl. *Java Server Pages*), Java Servlets i CGI (engl. *Common Gateway Interface*).

Sučelje opisano isključivo HTML jezikom nedostatno je za potrebe mnogih usluga. Kako bi se korisničkom sučelju dodale naprednije mogućnosti, HTML dokumentima dodaje korisnička logika. Korisničku logiku čine programski odsječci koji se izvode na korisničkom računalu, najčešće u sklopu Web preglednika. Njima se ostvaruje interaktivnost sučelja, animacije teksta i slike, te zvučni efekti. Dio posla koji bi inače morala obaviti poslužiteljska logika prebacuje se na korisničku logiku, te se time rasterećuju poslužitelji. Najčešće korištene tehnologije korisničke logike su VBScript, JavaScript, Java Applet, ActiveX i drugo.

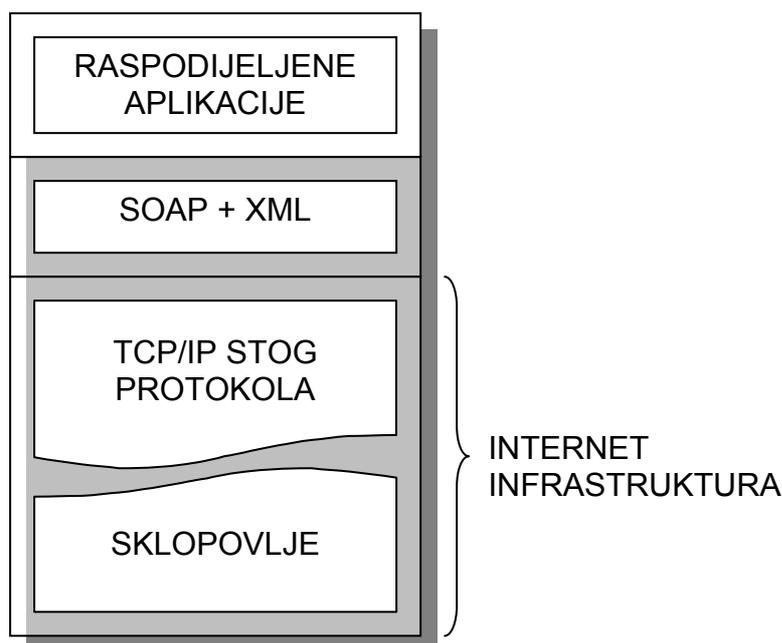
Razvijene tehnologije pojednostavile su razvoj usluga i omogućile njihovu brzu dostupnost. Kao rezultat, javlja se nagli porast broja raznovrsnih usluga. Web je postao najkorišteniji dio globalne mreže Internet, te se često pod imenom Internet zapravo misli upravo na Web. Danas je putem Weba dostupno mnoštvo novih usluga i neke od usluga koje otprije imaju svoje aplikacijske protokole. Tako se putem Weba danas nude usluge primanja i slanja elektroničke pošte (engl. *Web-mail*), usluge objavljivanja i čitanja novosti na Web forumima i druge. Od novih vrsta usluga najpoznatije su elektroničko bankarstvo (engl. *e-banking*), *on-line* aukcije, Web dućani, elektronička birokracija (engl. *e-government*), usluga pretraživanja Web prostora i drugo. Također, dio usluga koje su danas dostupne putem Weba prije su se nudile na neke druge načine. U početku je Web korišten samo kao izvor dodatnih informacija o uslugama, da bi se s vremenom same usluge polako premjestile na Web. Najbolji primjer za to su Web dućani. U početku su na Webu samo objavljivane informacije o nekom dućanu, da bi se s vremenom sama usluga prodaje preselila na Web.

2.2. Razvoj Web Services usluga

Od prvog standarda za poziv udaljenih procedura (RPC), objavljenog sredinom 80-ih godina prošlog stoljeća, razvijene su mnoge tehnologije namijenjene pozivu udaljenih procedura, kao što su DCOM (engl. *Distributed Component Object Model*), CORBA (engl. *Common Object Request Broker Architecture*), .NET Remoting ili Java RMI

(engl. *Remote Method Invocation*). Međutim, navedene tehnologije nisu uspjele udovoljiti zahtjevima uspostavljanja međuračunalne komunikacije na Internetu.

U međuvremenu, željeno pojednostavljenje razvoja usluga omogućio je Web. Web tehnologije uspjele su udovoljiti zahtjevima kojima nisu udovoljile prethodno navedene RPC tehnologije. Tako se javlja ideja da se na osnovu dotadašnjih Web tehnologija izgradi RPC infrastruktura. RPC tehnologija zasnovana na Webu nazvana je Web Services (slika 6). Činjenica da se zasniva na općeprihvaćenim Web tehnologijama pridonosi prihvaćanju Web Services tehnologije. Dodatno, u posljednje vrijeme razvojna okruženja na svim platformama pružaju neki oblik potpore oblikovanju raspodijeljenih aplikacija korištenjem Web Services tehnologije. Iako takva rješenja još uvijek nisu potpuno usklađena, može se reći da Web Services tehnologija predstavlja novi standard međuračunalne komunikacije.

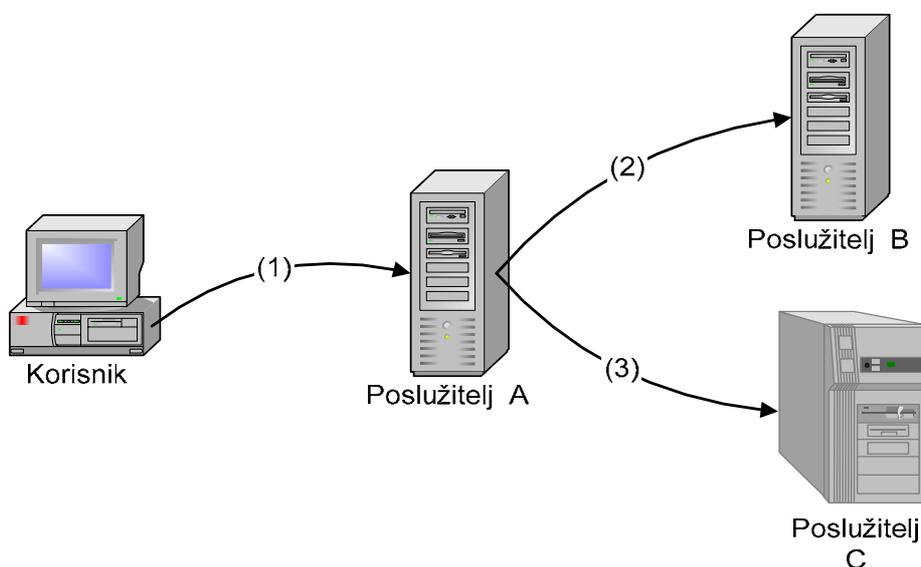


Slika 6. Smještaj sloja Web Services tehnologija na Internetu.

Tehnologija Web Services omogućava pozive udaljenih procedura korištenjem Web tehnologija. Pritom poslužiteljska logika preuzima ulogu udaljene procedure. Prilikom poziva Web usluge na korisničkoj strani nalazi se Web preglednik za kojim se obično nalazi krajnji korisnik. Kod Web Services usluga korisnička strana može biti bilo koji proizvoljni program. Iz tog razloga, odgovor poslužitelja Web Services usluge ne

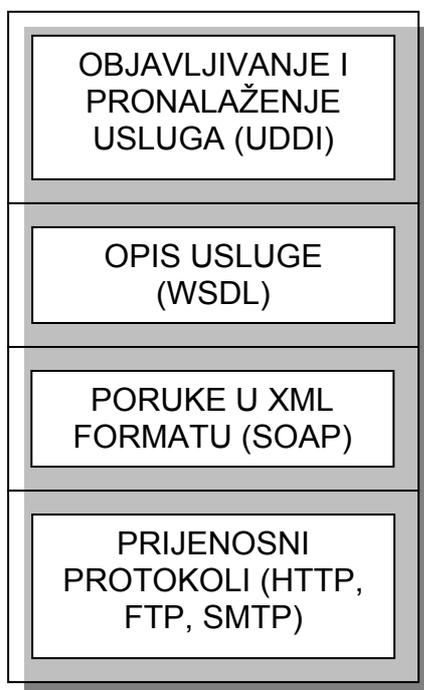
sadrži opis sučelja, te se tehnologije korisničke logike u Web Services tehnologiji ne koriste. Umjesto toga, u zahtjevima Web poslužitelju i njegovim odgovorima šalju se parametri udaljene procedure.

Prilikom poziva Web Services usluga korisnička strana može biti bilo koji proizvoljni program, pa tako i poslužitelj neke druge Web Services usluge. Kao rezultat, javlja se ulančavanje usluga. Na slici 7 prikazan je primjer ulančavanja Web Services usluga. Program na korisničkom računalu poziva Web Services uslugu A (1). Za izvođenje usluge A potrebni su podatci kojima se može pristupiti pomoću Web Services usluge koja se nalazi na računalu B, obrađeni na način koji omogućava Web Services usluga koja se nalazi na poslužitelju C. Zbog toga Web Services usluga A poziva Web Services uslugu B (2). Nakon završetka izvođenja usluge B, usluga A poziva uslugu C (3) s rezultatima izvođenja usluge B kao ulaznim parametrima. Nakon završetka izvođenja usluge C, usluga A završava nastavljajući s izvođenjem koristeći pritom rezultate izvođenja usluge C. Nakon završetka izvođenja usluge A, korisničkom programu vraćaju se rezultati izvođenja.



Slika 7. Komunikacija prilikom poziva Web Services usluge.

Web Services usluge ostvarene su korištenjem četiri tehnologije: XML [6], SOAP [7], WSDL [8] i UDDI [9], koje se nadovezuju jedna na drugu tvoreći konceptualni stog prikazan na slici 8.



Slika 8. Podjela Web Services tehnologije na slojeve.

XML (engl. *eXtensible Markup Language*) je tekstualni standard namijenjen opisu podataka, s posebnim naglaskom na njihovoj strukturi. Ne ovisi o sklopovskoj i programskoj potpori, što ga čini pogodnim za prijenos podataka između različitih računalnih platformi.

Web Services pozive udaljenih procedura ostvaruju korištenjem SOAP (engl. *Simple Object Access Protocol*) protokola. SOAP je jednostavni protokol, zasnovan na XML-u, namijenjen razmjeni poruka u raspodijeljenoj okolini. Za prijenos SOAP poruka mogu se koristiti različiti prijenosni protokoli, kao na primjer HTTP, FTP (engl. *File Transfer Protocol*), SMTP (engl. *Simple Mail Transfer Protocol*) i drugi. SOAP se najčešće koristi kao mehanizam za pozivanje udaljenih procedura. Budući da je pritom uobičajena komunikacija oblika zahtjev-odgovor, HTTP se pokazao pogodnim za prijenos SOAP poruka. Pritom su SOAP zahtjevi ugniježđeni u HTTP zahtjeve, a SOAP odgovori u HTTP odgovore. SOAP je detaljnije opisan u poglavlju 2.2.3.

WSDL (engl. *Web Services Description Language*) je jezik zasnovan na XML-u kojim se opisuje Web Services usluga. Opisuju se procedure koje se u sklopu usluge nude, način njihova poziva, ulazni parametri, povratne vrijednosti i slično. WSDL je detaljnije opisan u poglavlju 2.2.4.

Web Services usluge je moguće je jedinstveno identificirati korištenjem URL-a (engl. *Uniform Resource Locator*). Općeniti oblik URL-a je

protokol://poslužitelj[:pristup]/[staza],

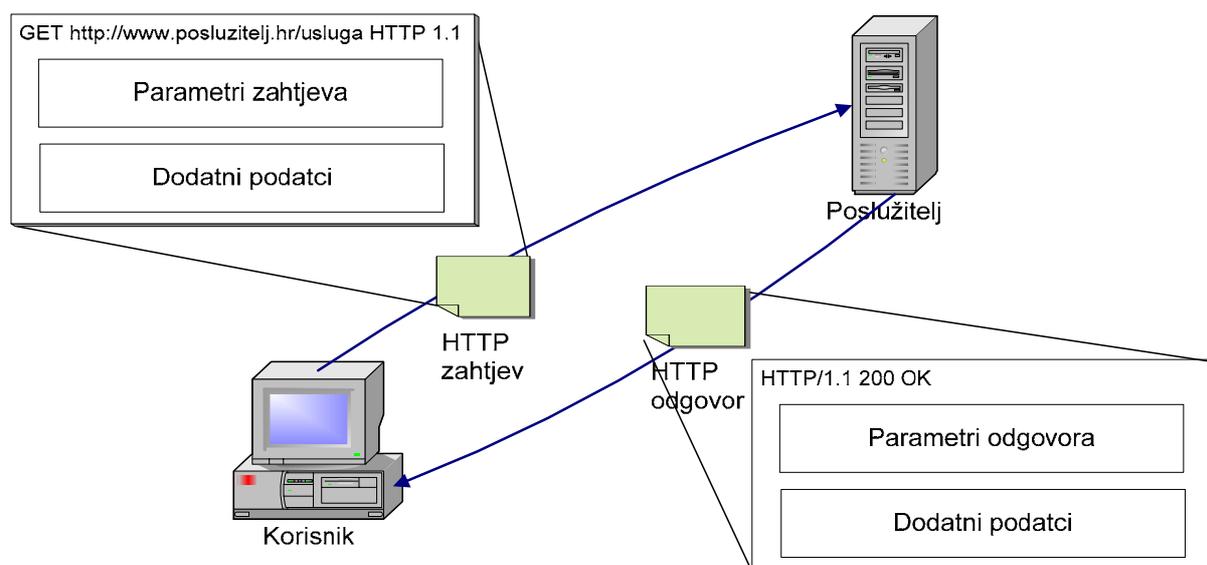
gdje je protokol naziv protokola (na primjer, ključna riječ *http* označava istoimeni protokol), poslužitelj ime ili IP (engl. *Internet Protocol*) adresa računala poslužitelja, pristup (engl. *port*) označava pristup na kojem poslužitelj prima zahtjeve navedenim protokolom, a staza određuje koja je od ponuđenih usluga na navedenom poslužitelju zatražena.

U slučaju kada ne postoji neposredni kontakt između poslužitelja neke Web Services usluge te njenog potencijalnog korisnika, javlja se potreba za mehanizmom objavljivanja i pronalaženja Web Services usluga. Takav mehanizam razvijen je po uzoru na Web pretraživače pomoću kojih se pretražuje Web prostor, ne bi li se pronašla Web stranica koja zadovoljava tražene kriterije. Pretraživači Web prostora ostvareni su kao Web usluge. Tako je i za potrebe objavljivanja i pronalaženja Web Services usluga razvijena specijalizirana Web Services usluga pod nazivom UDDI (engl. *Universal Description, Discovery and Integration*).

2.2.1. HTTP

Protokoli se mogu podijeliti na binarne i tekstualne. U porukama binarnih protokola sadržaj je sažeto zapisan čime se postiže učinkovitost komunikacije. Budući da je takav zapis teško čitati, razvoj i razumijevanje takvih protokola nije jednostavno. Za razliku od binarnih protokola, kod tekstualnih protokola smanjuje se učinkovitost radi postizanja veće jednostavnosti. HTTP je tekstualni protokol. Njegove su poruke čitljive i lako razumljive, što pojednostavljuje razvoj tehnologija i proizvoda zasnovanih na njemu.

U skladu s komunikacijskim modelom korisnik-poslužitelj, prema kojem je razvijen, razlikujemo dvije vrste HTTP poruka – zahtjeve i odgovore. HTTP zahtjev sastoji se od linije zahtjeva, parametara zahtjeva i dodatnih podataka. Linijom zahtjeva određuje se metoda zahtjeva i verzija protokola, te se identificira tražena usluga. Postoji više metoda zahtjeva od kojih se najčešće koriste metode GET i POST. Metoda GET koristi se kada je potrebno dohvatiti neki sadržaj s poslužitelja, dok se metoda POST koristi kada je potrebno poslati podatke od korisnika prema poslužitelju. Tražena usluga određena je svojim URL-om. Parametrima zahtjeva moguće je detaljnije specificirati zahtjev (npr. traže se podaci samo ako su promijenjeni u posljednjih 24 sata). HTTP zahtjev može sadržavati i dodatne podatke koji se šalju poslužitelju na obradu. Kod Web usluga to je skup dodatnih uvjeta koji detaljnije specificiraju traženu uslugu, a nisu predviđeni standardnim parametrima, dok se kod Web Services usluga tako prenose parametri pozvane procedure.



Slika 9. HTTP zahtjev i odgovor.

HTTP odgovor sastoji se od statusne linije odgovora, parametara odgovora i dodatnih podataka. Statusna linija odgovora nosi informaciju o tome je li zahtjev uspješno izveden, a u slučaju da nije, koji je razlog neuspjeha. Status se zapisuje u obliku troznamenkastog broja u dekadskom brojevnom sustavu uz kratko tekstualno objašnjenje. Tako se, na primjer, za uspješno izvedene zahtjeve koriste brojevi kojima je prva znamenka 2, za pogreške klijenta brojevi kojima je prva znamenka 4,

dok se za pogreške poslužitelja koriste brojevi sa znamenkom 5 na prvom mjestu. Parametri odgovora sadrže dodatne informacije o dohvaćenim podacima (npr. trenutak zadnje izmjene). Kod Web usluga dodatni podatci predstavljaju opis sučelja, a kod Web Services usluga povratnu vrijednost pozvane procedure.

2.2.2. XML

XML (engl. *eXtensible Markup Language*) je jednostavan i fleksibilan jezik razvijen za opis strukturiranih podataka. Razvijen je od strane W3C-a (engl. *World Wide Web Consortium*), udruge koja se bavi izradom standarda namijenjenih upotrebi u Web okruženju. Od objavljivanja prve specifikacije, u veljači 1998. godine, XML se razvio u najkorišteniji standard za razmjenu strukturiranih podataka na Internetu.

Slično većini standarda na Webu, XML je tekstualni standard. Iako ga to čini manje učinkovitim, pridonosi njegovoj jednostavnosti i pristupačnosti. Sintaksa jezika XML vrlo je slična sintaksi jezika HTML. Zasniva se na oznakama (engl. *tag*). Za razliku od HTML-a u XML-u ne postoje unaprijed definirane oznake, već autor svakog dokumenta može kreirati vlastite oznake. Zbog tog svojstva XML je nazvan proširivim (engl. *extensible*). Dodatna razlika između ta dva jezika je u njihovoj namjeni. Dok se jezik HTML koristi za definiranje načina prikaza podataka, XML se pokazao pogodnim za opis podataka, s posebnim naglaskom na njihovoj strukturi. Budući da je nezavisan o sklopovskoj i programskoj potpori, jezik XML pogodan je za prijenos podataka između različitih računalnih platformi.

Osnovni građevni blokovi XML dokumenata su XML elementi. Postoje dvije vrste elemenata – elementi sa sadržajem i prazni elementi. Elementi sa sadržajem sastoje se od početne oznake, koja eventualno može sadržavati dodatne attribute, sadržaja, te završne oznake, kao na primjer:

```
<element_sa_sadržajem atribut="vrijednost">sadržaj</element_sa_sadržajem>.
```

Prazni elementi nemaju sadržaj ni završnu oznaku. Kako bi ih se moglo razlikovati od početnih oznaka elemenata sa sadržajem, na kraju imaju kosu crtu (/), kao na primjeru:

```
<prazni_element atribut="vrijednost"/>.
```

Jezik XML posebno je pogodan za prikaz hijerarhijski organiziranih podataka, zbog mogućnosti ulančavanja elemenata. Na slici 10 je prikazan primjer XML dokumenta na kojem se vidi kako se elementi ulančavaju.

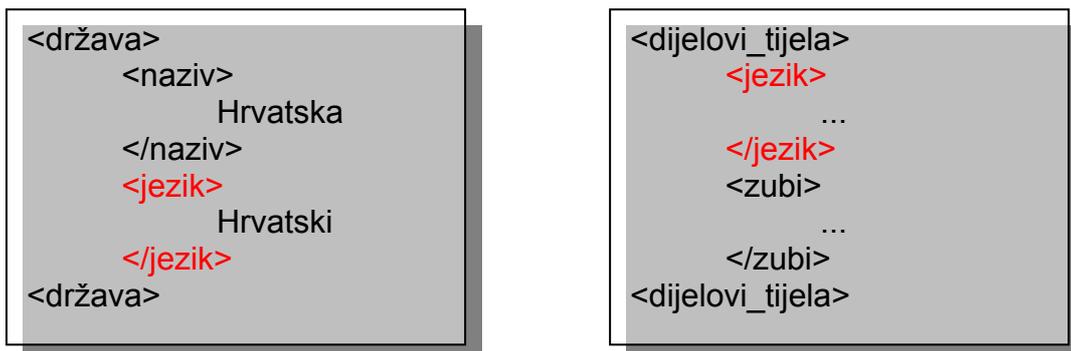
```
<adresa>
  <osobni_podaci>
    <ime> Ivor </ime>
    <prezime> Sučić </prezime>
  </osobni_podaci>
  <ulica> Račkoga 2a </ulica>
  <grad> Zagreb </grad>
</adresa>
```

Slika 10. Primjer odsječka XML dokumenta.

Na XML dokumente postavljaju se dva zahtjeva. Prvi zahtjev je da dokument mora biti ispravno oblikovan (engl. *well formed*). XML dokument smatra se ispravno oblikovanim ako svaka početna oznaka XML elementa sa sadržajem ima svoju završnu oznaku, te ako su elementi ispravno ulančani.

Drugi zahtjev je da dokument mora biti valjan (engl. *valid*). Budući da jezik XML nema unaprijed definiranu semantiku, kao ni unaprijed definirani skup oznaka, razvijene su standardne metode za definiranje skupova oznaka, dodatnih pravila ulančavanja, korištenja atributa i slično. Starija metoda korištenja DTD (engl. *Document Type Definition*) dokumenata s dodatnim pravilima sve se češće zamjenjuje korištenjem XML Schema dokumenta. Za razliku od DTD dokumenata koji su pisani posebnim jezikom, XML Schema dokumenti pisani su jezikom XML, čime se otklanja potreba učenja dodatnog jezika. XML dokument koji zadovoljava pravila zadana odgovarajućim DTD ili XML Schema dokumentom smatra se valjanim.

Uz mnoge prednosti, mogućnost kreiranja proizvoljnih imena elemenata ima i jedan nedostatak. Moguće je da isti naziv elementa ima različitu semantiku ovisno o kontekstu u kojem se upotrebljava, što može izazvati probleme nejednoznačnosti. Na slici 11 prikazana su dva odsječka XML dokumenta koji sadrže elemente istog naziva, a različite semantike. U prvom primjeru riječ „jezik“ ima značenje jezika kojim se govori u nekoj državi, dok u drugom primjeru ista riječ ima značenje dijela tijela.



Slika 11. Primjer nejednoznačnosti XML elemenata.

Kako bi se takve nejednoznačnosti izbjegle uveden je pojam imenika (engl. *namespace*). Imenici se koriste na način da se za svaki element, osim njegovog naziva, specificira i imenik kojem taj naziv pripada. Pretpostavka je da u jednom imeniku jedan naziv može imati samo jedno značenje. Kako bi nazivi imenika bili jedinstveni širom Interneta, za njihovo imenovanje koristi se URI (engl. *Uniform Resource Identifier*). URI je jedinstveni niz znakova koji služi za identifikaciju apstraktnog ili fizičkog entiteta, te omogućava jedinstvenost podataka. Primjer URI-ja je URL, spomenut u poglavlju 2.2.

Budući da bi pisanje naziva imenika pored naziva svakog XML elementa činilo XML dokumente nepreglednima, uvodi se mogućnost određivanja kratice koja će predstavljati imenik. Kratica se definira unutar oznake za početak XML elementa, slično kao XML atribut, na primjer:

```
<naziv_elementa xmlns:kratica="imenik">.
```

Kratica, kao zamjena za imenik, vrijedi samo u elementu u kojem je definirana i u elementima koji su u hijerarhijskom stablu njemu podređeni. Kratica se koristi na način da se ispred naziva elementa navede prvo kratica, a zatim naziv elementa odvojeni dvotočkom, kao na primjer:

```
<kratica:naziv_elementa>.
```

Ovakvim zapisom označava se da navedeni element spada u imenik koji navedena kratica predstavlja.

U slučaju kada nazivi svih elemenata nekog dijela hijerarhijskog stabla pripadaju istom imeniku, pogodno je imenik deklarirati implicitno. Imenik se deklarira bez navođenja kratice, kao na primjer:

```
<naziv_elementa xmlns="imenik">
```

Nakon implicitne deklaracije podrazumijeva se da nazivi svih elemenata podređenog hijerarhijskog stabla, koji nemaju eksplicitno navedenu kraticu, pripadaju implicitno deklariranom imeniku.

2.2.3. SOAP

SOAP (engl. *Simple Object Access Protocol*) je protokol kojim su formalno definirane standardizirane XML poruke za razmjenu informacija u raspodijeljenim sustavima. Konceptualno, SOAP poruka zamišljena je kao pismo. Sastoji se od zaglavlja (engl. *header*) i tijela (engl. *body*) dokumenta, omotanih omotnicom (engl. *envelope*), kao što je prikazano na slici 12. U stvarnosti, SOAP poruka je XML dokument čiji je korijenski element nazvan „Envelope“, a njegova dva podelementa „Header“ i „Body“. Oni su, kao i svi ostali nazivi elemenata definiranih SOAP protokolom, dio imenika „<http://www.w3.org/2003/05/soap-envelope>“.



Slika 12. SOAP omotnica.

SOAP je zamišljen kao fleksibilan protokol za razmjenu poruka. Stoga, SOAP omogućuje slanje poruka od inicijalnog pošiljatelja do krajnjeg primatelja posredstvom više SOAP posrednika (engl. *SOAP intermediaries*). Svaki od posrednika može pružati neku dodatnu uslugu (engl. *value-add service*), primjerice usmjeravanja poruke, kriptiranja ili slično. Potrebno je razlikovati podatke namijenjene posrednicima od onih namijenjenih isključivo krajnjem primatelju poruke. Posrednici mogu pregledavati, dodavati, brisati ili mijenjati isključivo one podatke koji su njima namijenjeni.

Tijelo je dio SOAP poruke koji služi prijenosu osnovnih informacija od inicijalnog pošiljatelja do krajnjeg primatelja poruke. Podatke koji se ondje nalaze posrednici ne smiju mijenjati niti brisati, niti smiju dodavati nove podatke u taj dio poruke. Budući da je tijelo poruke njezin osnovni dio, ono se obavezno mora nalaziti u svakoj SOAP poruci.

Zaglavlje je dio SOAP poruke koji predstavlja mehanizam za prijenos kontrolnih podataka. To ga čini pogodnim za smještaj podataka namijenjenih SOAP posrednicima. Kontrolni podatci, smješteni u zaglavlju, mogu se upotrebljavati za prilagođavanje SOAP poruka specifičnim zahtjevima neke aplikacije. U takvim slučajevima oni nisu namijenjeni posrednicima već krajnjem primatelju. Na primjer, u zaglavlju se mogu prenositi podatci za potrebe autentikacije (engl. *authentication*). Budući da se u zaglavlju prenose podatci koji izlaze iz osnovnog sadržaja SOAP poruke, ono je neobavezni dio poruke.

Način na koji bi se zaglavlje SOAP poruke koristilo u navedene svrhe nije standardiziran. Zato je u slučaju kada se ga se želi koristiti potreban prethodni dogovor svih strana u komunikaciji SOAP protokolom, što često nije moguće. Osim toga, za usmjeravanje SOAP poruka obično se brine neki od protokola niže razine, tako da sa stanovišta SOAP poruke ona putuje direktno od inicijalnog pošiljatelja do krajnjeg primatelja. Upravo iz tih razloga, u praksi se zaglavlje SOAP poruka rijetko koristi.

Osim građe samih poruka, SOAP-om su standardizirane i metode njihove razmjene. To je ostvareno uvođenjem konvencija o komunikaciji SOAP porukama oslanjanjem

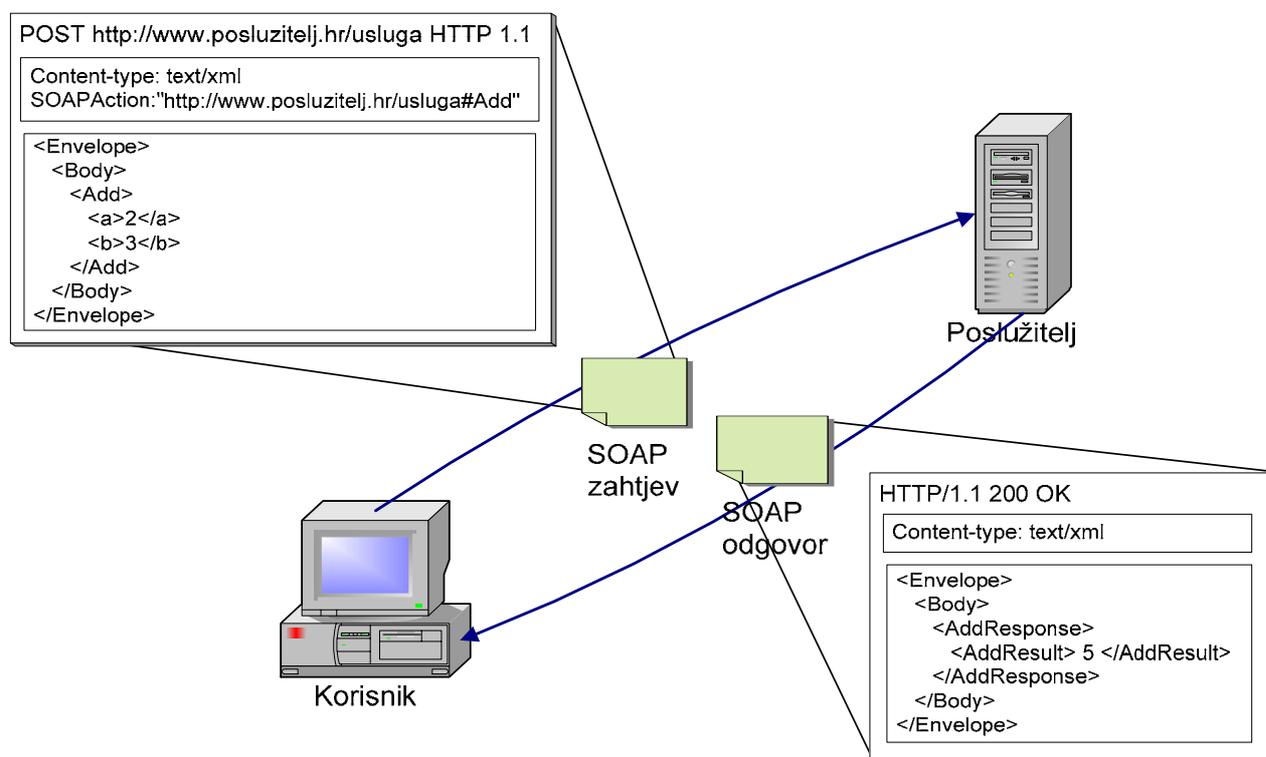
na prijenosne protokole (engl. *SOAP bindings*). Za svaki pojedinačni prijenosni protokol uvodi se poseban skup pravila koji opisuje kako ga se može koristiti za razmjenu SOAP poruka. Od prijenosnih protokola, kao što su UDP (engl. *User Datagram Protocol*) i TCP (engl. *Transmission Control Protocol*), za prijenos SOAP poruka mnogo se češće koriste, već spomenuti, protokoli aplikacijske razine kao što su FTP i SMTP, a najčešće HTTP.

U kombinaciji s navedenim prijenosnim protokolima, te specifičnim zahtjevima aplikacija koje komuniciraju, korištenjem SOAP protokola moguće je izgraditi razne komunikacijske modele. Tako se za pozive udaljenih procedura ostvarenih u Web Services tehnologiji koristi komunikacijski model zahtjev-odgovor izgrađen kombinacijom SOAP-a i HTTP-a. Iako standard ne propisuje upotrebu HTTP-a kao prijenosnog protokola, njegova je upotreba postala *de facto* standardom. Shematski prikaz poziva udaljene procedure ostvarene u Web Services tehnologiji prikazan je na slici 13.

Na zahtjev aplikacije koja poziva udaljenu proceduru, komunikacijski posrednik na računalu klijentu stvara SOAP poruku koja predstavlja zahtjev za pokretanjem procedure. U tijelo navedene poruke smještaju se parametri poziva procedure pretvoreni u XML oblik. Prema konvenciji, parametri poziva procedure objedinjavaju se XML elementom nazvanim prema imenu pozvane procedure. Poruka se zatim smješta u HTTP zahtjev koji se šalje SOAP poslužitelju. Osim prijenosne uloge HTTP ovdje preuzima i ulogu identifikacije krajnjeg primatelja SOAP poruke. URI SOAP čvora kojem je poruka namijenjena smješta se u jedan od parametara HTTP zahtjeva (parametar „SOAPAction“). Na poslužitelju, komunikacijski posrednik iz primljene poruke izdvaja ulazne parametre pozvane procedure. Pokreće se izvođenje procedure, te joj se predaju ulazni parametri pretvoreni iz XML oblika u potreban oblik, ovisno o tehnologiji u kojoj je procedura ostvarena.

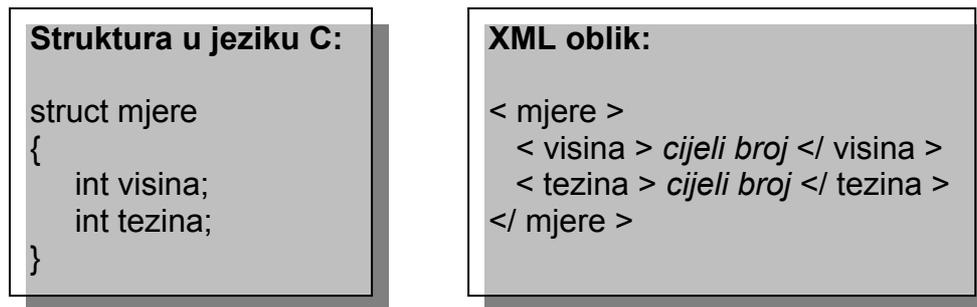
Nakon završetka izvođenja procedure, komunikacijski posrednik na poslužitelju stvara novu SOAP poruku. Izlazni parametri, pretvoreni u XML oblik, smještaju se u tijelo navedene poruke. Prema konvenciji, povratni parametri objedinjavaju se XML elementom čiji se naziv tvori od imena udaljene procedure i riječi „Response“. U slučaju da prilikom obrade zahtjeva na poslužitelju dođe do pogreške, u tijelo poruke

koja predstavlja odgovor smještaju se podaci o pogrešci. SOAP poruka se zatim smješta u HTTP odgovor, te se šalje nazad računalu koje je poslalo zahtjev. Na računalu klijentu izlazni se parametri iz XML oblika pretvaraju u oblik koji zahtijeva tehnologija u kojoj je izvedena aplikacija koja je pozvala udaljenu proceduru. U tom se obliku zatim vraćaju aplikaciji. Sa stanovišta pozivajuće aplikacije nema razlike nalazi li se pozvana procedura na istom računalu ili na nekom drugom udaljenom računalu jer posao vezan uz komunikaciju s poslužiteljem obavlja komunikacijski posrednik.



Slika 13. Poziv udaljene procedure korištenjem SOAP i HTTP protokola.

SOAP protokol standardizira detalje zapisa parametara pozvane procedure u tijelu poruke. Ipak, parametre je dozvoljeno zapisati i na neki korisnički definirani način. Na slici 14 prikazana je deklaracija strukture podataka napisana u jeziku C, te njezin XML oblik izrađen prema standardnim SOAP pravilima serijalizacije.



Slika 14. Primjer serijalizacije podatkovne strukture, pisane jezikom C, u skladu sa SOAP protokolom.

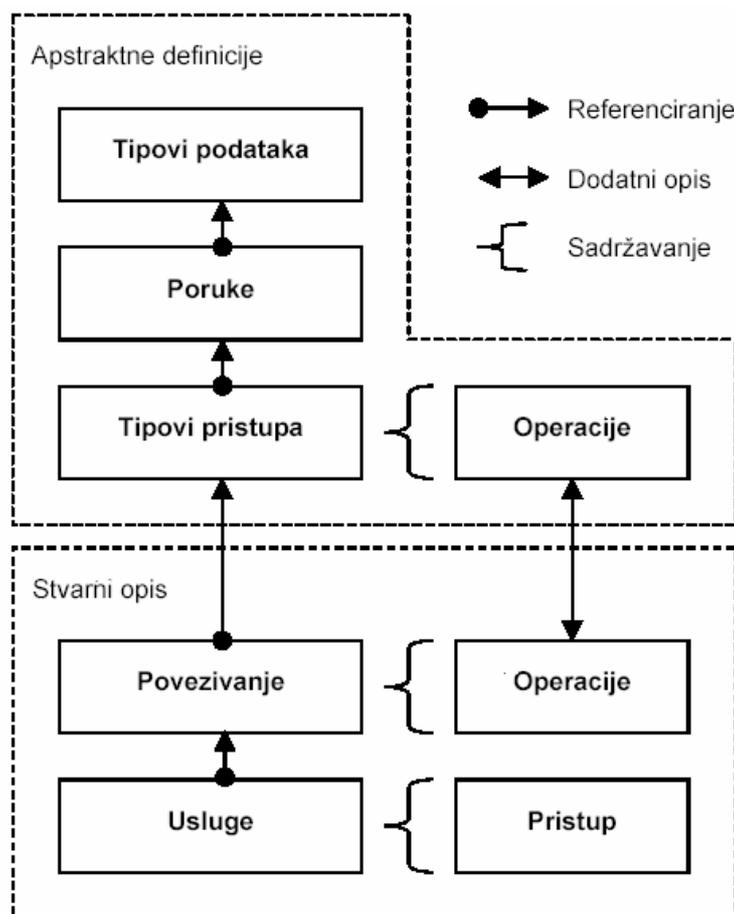
2.2.4. WSDL

Web Services usluga obično se sastoji od mnoštva udaljenih procedura koje se mogu pozivati korištenjem dosad opisanih protokola. Budući da korisnički i poslužiteljski dio aplikacije najčešće razvijaju različiti programeri, potreban je mehanizam kojim bi se informacije potrebne za pravilni poziv udaljenih procedura učinili javno dostupnima. Iz tog razloga svaka Web Services usluga opisuje se pomoću jednog ili više WSDL (engl. *Web Services Description Language*) dokumenata. WSDL dokumenti najčešće su dostupni na poslužitelju usluge koju opisuju.

WSDL je jezik za opis Web Services usluge zasnovan na XML-u. Usluge se opisuju navođenjem ponuđenih procedura i njihovih parametara, te opisom potrebnih tipova podataka i predviđenih prijenosnih protokola.

U WSDL dokumentima usluge (engl. *service*) se prikazuju kao skupovi pristupnih točaka (engl. *port*), od kojih svaku čini skup procedura (engl. *operation*). Slično kao što se srodne funkcije običavaju objediniti u biblioteke funkcija, tako se u ovom slučaju preporuča srodne procedure objediniti u pristupnu točku. Na ovaj način uslugu se može promatrati kao skup biblioteka udaljenih procedura. Ovakvom hijerarhijskom organizacijom poboljšana je preglednost, te pojednostavljeno pronalaženje potrebne procedure.

WSDL dokument može se podijeliti na dvije osnovne logičke cjeline – apstraktne definicije i konkretne opise – kao što je prikazano na slici 15. Apstraktnom se smatra svaka definicija kojom se ne specificira način pristupa usluzi, odnosno protokol i adresa. Apstraktna definicija dodatno se dijeli na tri dijela, po jedan za opis tipova podataka, poruka (engl. *message*) koje se razmijenjuju prilikom poziva procedure, te tipova pristupnih točaka (engl. *port types*).



Slika 15. WSDL dokument.

Tipovi podataka opisuju se nezavisno od bilo kojeg programskog jezika, prema istom principu kao i u XML Schema dokumentima. Na isti način kao što se u XML Schema dokumentima definiraju pravila za XML dokumente, tako se u WSDL dokumentima definiraju pravila za SOAP poruke. Korištenjem unaprijed definiranih osnovnih tipova podataka, kao što su, na primjer, cijeli brojevi (engl. *integer*) ili znakovni nizovi (engl. *string*), moguće je definirati razne složene podatkovne strukture.

Poruke se u WSDL dokumentima apstraktno definiraju kao nizovi podatkovnih objekata. Svakom podatkovnom objektu odgovara jedan tip definiran u za to predviđenom, gore opisanom dijelu.

Tip pristupne točke definira se na način da se navode sve procedure, te, za svaku od njih, referencira najviše po jedna definicija poruke za ulaznu i izlaznu poruku.

Odnos između konkretnih opisa i apstraktnih definicija u WSDL-u može se usporediti s odnosom između instanciranih objekata i definicija razreda u objektno orijentiranom dizajnu. Konkretni opisi sadrže informacije o konkretnim uslugama referencirajući se na apstraktne definicije. Sastoje se od dva dijela. Cjelina **povezivanja** (engl. *binding*) referencira jedan tip pristupne točke, te za svaku pojedinu proceduru definira detalje poziva – URI koji jedinstveno identificira proceduru i prijenosni protokol prilikom poziva. Cjelina **usluge** sadrži internetsku adresu poslužitelja na kojem je dostupna, popis pripadajućih pristupnih točaka, te referencu na odgovarajuću cjelinu povezivanja za svaku od njih.

Ovakvom podjelom u dijelove s mogućnošću referenciranja, te kombinacijom apstraktnih definicija i konkretnih opisa, izbjegava se potreba za višestrukim definiranjem istih tipova podataka, poruka ili tipova pristupnih točaka. Štoviše, pojedini dijelovi mogu se nalaziti u različitim WSDL dokumentima.

2.3. Proširenja Web Services tehnologije

Objektno orijentirani pristup često se koristi u programerskoj praksi. U programiranju raspodijeljenih aplikacija umjesto pozivanja udaljenih procedura, objektno orijentirani pristup donosi mogućnost pozivanja udaljenih objekata (engl. *remote object invocation*). Razlika je u tome što je kod objektno orijentiranog programiranja predviđeno da se na poslužitelju pamti stanje izvođenja, dok kod proceduralnog programiranja to nije slučaj. Stanje izvođenja predstavljaju trenutne vrijednosti varijabli korištenih pri izvođenju pozvane procedure ili objekta. Pozivi udaljene procedure međusobno su nezavisni. Stanje izvođenja gubi se na kraju izvođenja procedure, te ga nije moguće zadržati između dva poziva. Za razliku od toga, kod

poziva udaljenih objekata moguće je pamtiti stanje izvođenja. To znači da objekt ima mogućnost zadržati stanje izvođenja, sve dok ga se ponovo ne pozove. Tada se za nastavak izvođenja koriste podaci zapamćeni prilikom prethodnog izvođenja. Mogućnost pamćenja stanja izvođenja pojednostavljuje razvoj raspodjeljenih aplikacija. Iz tog razloga objektno orijentirani pristup smatra se pogodnijim od proceduralnog za razvoj raspodijeljenih aplikacija.

Web Services tehnologija zasniva se na proceduralnom pristupu. Kako bi se unaprijedilo njezine mogućnosti, te se ona učinila sličnijom svojstvima objektno orijentiranih tehnologija, javlja se želja za omogućavanjem pamćenja stanja izvođenja. Uvođenjem mogućnosti pamćenja stanja izvođenja javlja se problem suprotstavljenih zahtjeva. Taj problem sličan je problemu koji se javlja kod dijeljenja spremnika (engl. *shared memory*). Kod dijeljenja spremnika, više korisnika pristupa istoj varijabli čime se narušava integritet zapisanih podataka. Kod pamćenja stanja izvođenja, više korisnika pristupa istom stanju izvođenja. U slučaju kada zahtjevi korisnika nisu međusobno povezani takva situacija najčešće nije poželjna. Kako bi se izbjegle takve situacije potrebno je omogućiti istovremeno pamćenje više zasebnih stanja izvođenja usluge.

Iako se Web Services standard zasniva se na proceduralnom pristupu, tehnologije ostvarenja Web Services usluga omogućavaju ostvarenje usluga koje pamte stanje izvođenja. Standardom nije predviđena mogućnost pamćenja stanja izvođenja, te ne postoje standardni načini razlikovanja zasebnih stanja izvođenja. Kako bi se Web Services tehnologija mogla koristiti za izgradnju usluga koje pamte stanje izvođenja uvode se proširenja Web Services standarda. Objavljena su dva prijedloga proširenja Web Services standarda – OGSi (engl. *Open Grid Services Infrastructure*) i WS-resources. Najznačajnije razlike među njima su kontekst njihova nastanka, te način razlikovanja zasebnih stanja izvođenja. U sljedećim poglavljima opisane su obje navedene tehnologije.

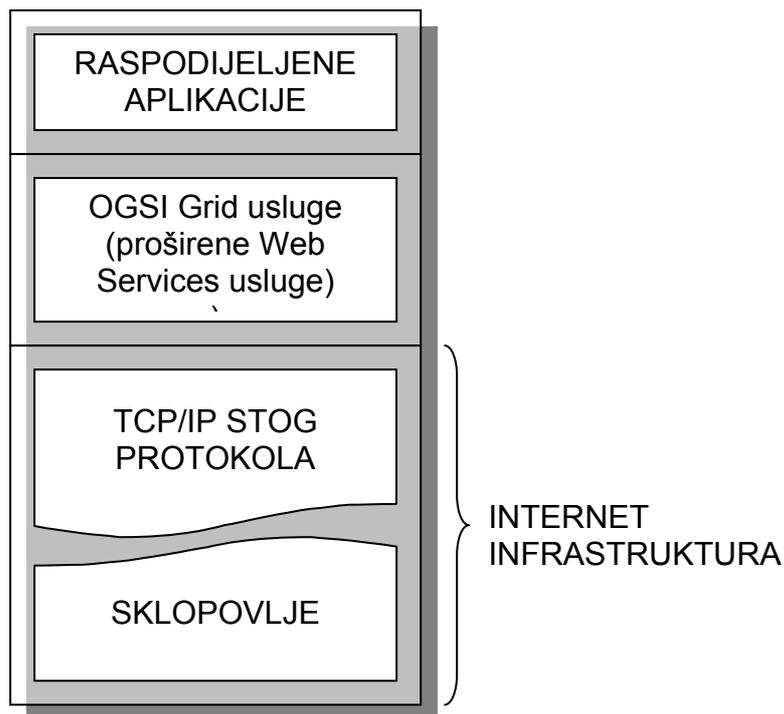
2.3.1. Grid usluge

Prvi prijedlog proširenja Web Services standarda nastao je u Grid zajednici pod nazivom OGSi [10]. Web Services usluge proširenih funkcionalnosti definirane OGSi standardom nazivaju se Grid usluge. U nastavku pogavlja objašnjen je kontekst nastanka OGSi standarda, te princip rada Grid usluga.

Nastanak OGSi standarda u Grid okruženju

Potreba za velikom računalnom snagom, u obliku brzine procesora, te količine prostora za čuvanje podataka, najprije se javila u znanstvenim krugovima. Kao rješenje, javlja se želja da se unaprijeđenjem Interneta stvori okruženje u kojem bi bila moguća razmjena računalnih kapaciteta diljem svijeta, tzv. Grid [11] [12]. Cilj uspostave Grid okruženja je omogućavanje razvoja raspodijeljenih aplikacija što sličnije načinu na koji se danas razvijaju „obične“ aplikacije. Takvo okruženje, osim standardnih protokola, uključuje i standardna aplikacijska sučelja (engl. *Application Programming Interface*), koji sadrže svu potrebnu komunikacijsku logiku, kako programeri ne bi morali brinuti o detaljima komunikacije s poslužiteljima.

Budući da je Grid zamišljen kao globalno okruženje, postoji snažni pritisak da se sve zasniva na općeprihvaćenim standardima, kao i da svi novodoneseni standardi budu općeprihvaćeni. Web Services tehnologija prihvaćena je kao osnova za razvoj Grid okruženja zbog svoje otvorenosti i općeprihvaćenosti, te zbog toga što učinkovito ostvaruje neke od zadanih ciljeva. Međutim, Web Services tehnologija ne zadovoljava sve potrebe uspostave Grid okruženja. Kako bi bolje udovoljila zahtjevima izgradnje Grid okruženja, Web Services tehnologija proširuje se OGSi standardom. OGSi standardom definirane su usluge proširenih funkcionalnosti nazvane Grid uslugama. Vizija Grid okruženja ostvarenog korištenjem Grid usluga nazvana je OGSA [13] (engl. *Open Grid Services Architecture*) (slika 16).



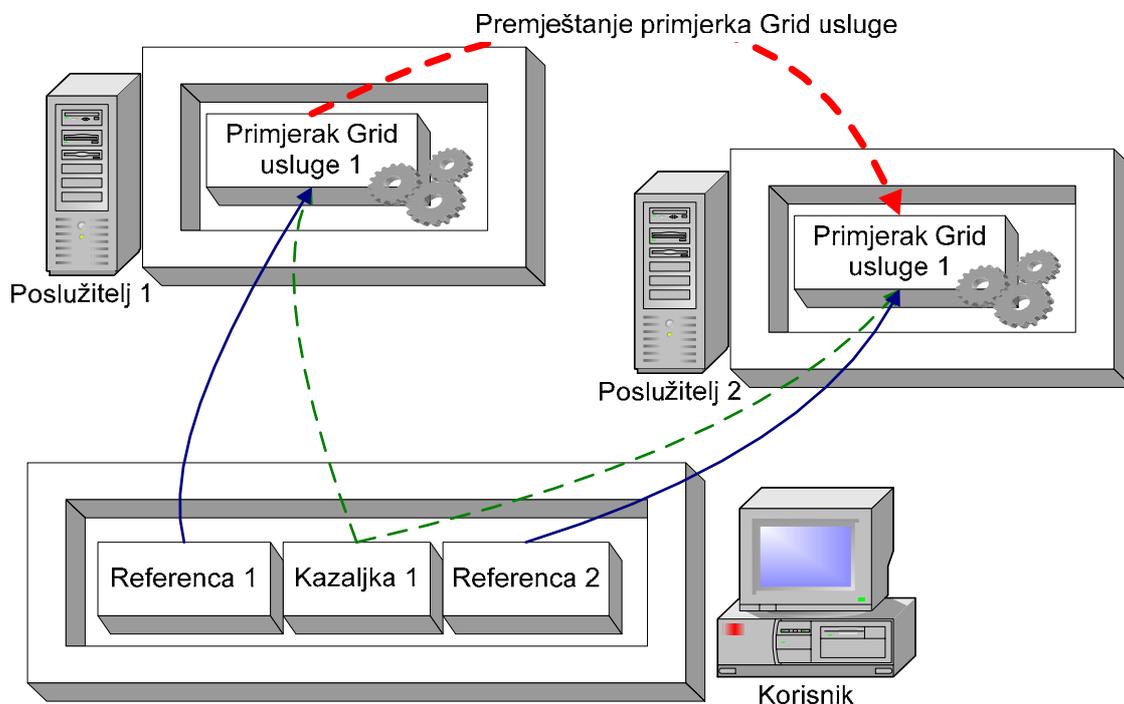
Slika 16. Slojevita arhitektura OGSA okruženja.

Princip rada Grid usluga

Za razliku od Web Services usluga, Grid usluge predviđene su da mogu pamtit stanje izvođenja. U skladu s time uveden je standardni način razlikovanja zasebnih stanja izvođenja usluge. Prema standardu, zasebna stanja izvođenja usluge razlikuju se korištenjem različitih primjeraka (engl. *instance*) iste Grid usluge. Istovremeno može postojati više primjeraka iste usluge od kojih svaki pamti po jedno zasebno stanje izvođenja. Stanje se čuva u varijablama stanja (engl. *service data element*) primjerka Grid usluge. Kako bi se varijable stanja mogle opisati, WSDL standard proširen je potrebnim elementima.

Svaki korisnik može stvoriti svoj primjerak usluge s isključivim pravima pristupa. Budući da jednom primjerku usluge pristupa samo jedan korisnik, osigurano je da se pri svakom sljedećem pozivu istog primjerka usluge izvršavanje nastavlja na mjestu gdje je posljednji put zaustavljeno. Svaki primjerak Grid usluge jednoznačno je i trajno definiran svojom kazaljkom (engl. *handle*). Kako bi se osigurala njezina jedinstvenost zadano je da kazaljka mora biti valjani URI. Kazaljka se svakom primjerku usluge dodjeljuje prilikom njezinog stvaranja. Korisnik koji je stvorio novi

primjerak usluge može ga od ostalih primjeraka iste usluge razlikovati korištenjem kazaljke.



Slika 17. Metoda pristupa primjerku Grid usluge.

Standard predviđa mogućnost da se primjerak Grid usluge u nekom trenutku premjesti s poslužitelja na poslužitelj (slika 17), odnosno da promjeni okolinu izvođenja (engl. *hosting environment*). Kazaljka Grid usluge dovoljno je općenita da omogući adresiranje odgovarajućeg primjerka usluge bez obzira na trenutnu okolinu izvođenja. Međutim, kako bi zadovoljila zahtjev za općenitošću, kazaljka ne može sadržavati dovoljno informacija da bi korisnik pomoću nje mogao pristupiti svojem primjerku usluge. Primjer informacije potrebne za pristup nekom primjerku usluge je informacija o komunikacijskom protokolu kojim mu se pristupa u trenutačnoj okolini izvođenja. Budući da kazaljka ne sadrži dovoljno informacija za pristup usluzi, uvodi se referenca (engl. *reference*) Grid usluge. Referenca Grid usluge detaljan je dokument sa svim podacima potrebnima za pristup nekom primjerku Grid usluge. Primjer reference je WSDL dokument. Referenca primjerka Grid usluge dinamički se stvara u ovisnosti o platformi na kojoj se instanca usluge trenutno nalazi. Zbog toga ona ne može biti trajna kao što je to kazaljka. Kada se primjerak usluga preseli na drugi poslužitelj referenca prestaje vrijediti.

2.3.2. WS-Resources

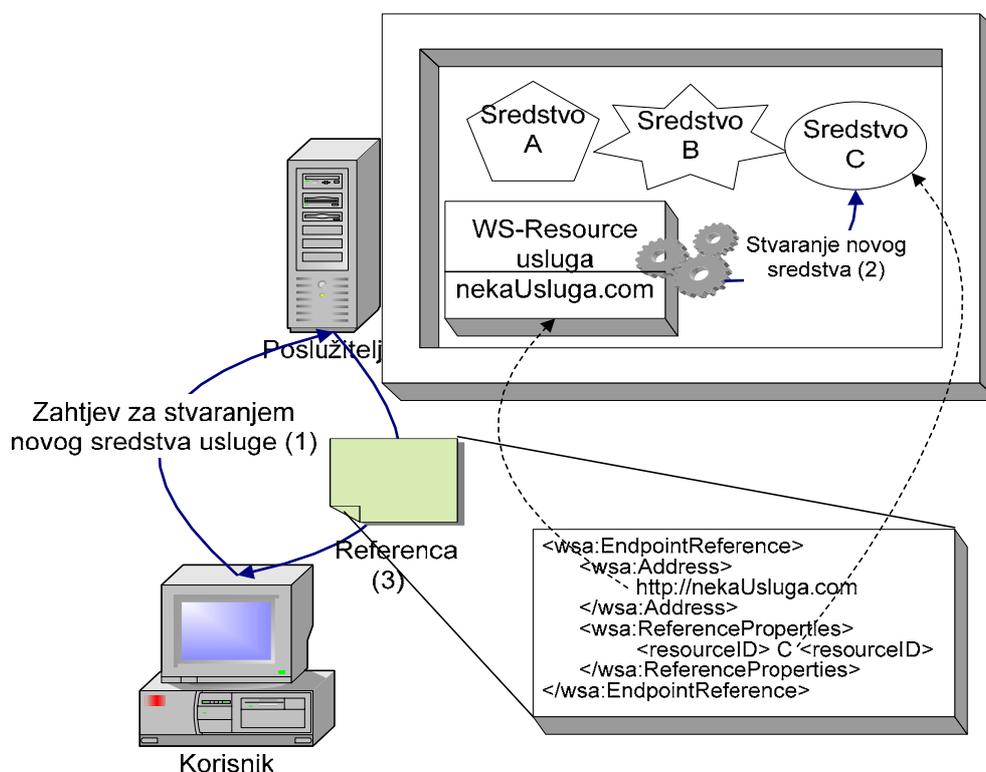
Ideja proširivanja Web Services standarda, kako bi se omogućila izgradnja usluga koje pamte stanje izvođenja javila se najprije u Grid zajednici. Koncepti predstavljeni u sklopu OGSi standarda nisu specifični samo za potrebe razvoja Grid okruženja, već omogućuju izgradnju općenitih uslugama orijentiranih arhitektura (engl. *service-oriented architecture*). Zbog toga se, s vremenom, javlja interes da se neki koncepti predstavljeni u sklopu OGSi standarda uvedu i u Web Services standard. Kao rezultat, razvijaju se razni dodatci Web Services standardu (npr. WS-Addressing). Razvojem navedenih dodataka Web Services standard dobiva mogućnosti slične onima definiranim OGSi standardom. WS-Resources [14] standard (engl. *WS-Resource framework*) nasljeđuje ideje i koncepte razvijene u sklopu OGSi standarda koristeći u tu svrhu najnovije dodatke Web Services tehnologiji. Budući da se razvija uz potporu gotovo svih vodećih tvrtke informatičke industrije, očekuje se prihvaćanje WS-Resources standarda.

Prema WS-Resources standardu usluge mogu pamti stanje izvođenja slično kao i Grid usluge. Razlika između Grid usluga i WS-Resources usluga je u načinu razlikovanja zasebnih stanja izvođenja usluga. Grid usluge pamte zasebna stanja izvođenja u zasebnim primjercima iste usluge. Za razliku od navedenog, WS-Resources usluge imaju samo po jedan primjerak, a zasebna stanja izvođenja pamte u zasebnim sredstvima usluge (engl. *resource*). Ono što su u Grid uslugama varijable stanja u WS-Resources uslugama postaju svojstva sredstva usluge (engl. *resource properties*). Oni se opisuju u zasebnom XML Schema dokumentu koji se zatim povezuje s WSDL dokumentom kojim je usluga opisana.

Svi korisnici pristupaju istom primjerku neke usluge. Kako bi se razlikovalo stanje izvođenja potrebno je pri pozivu usluge specificirati koje sredstvo usluge koristiti pri izvođenju. Za prijenos informacije o tome koje sredstvo usluge je potrebno koristiti pri izvođenju usluge koristi se zaglavlje SOAP poruke. Način na koji se navedena informacija prenosi u zaglavlju SOAP poruke specificiran je WS-Addressing [16] standardom.

Upotreba WS-Addressing standarda za razlikovanje zasebnih stanja izvođenja usluge

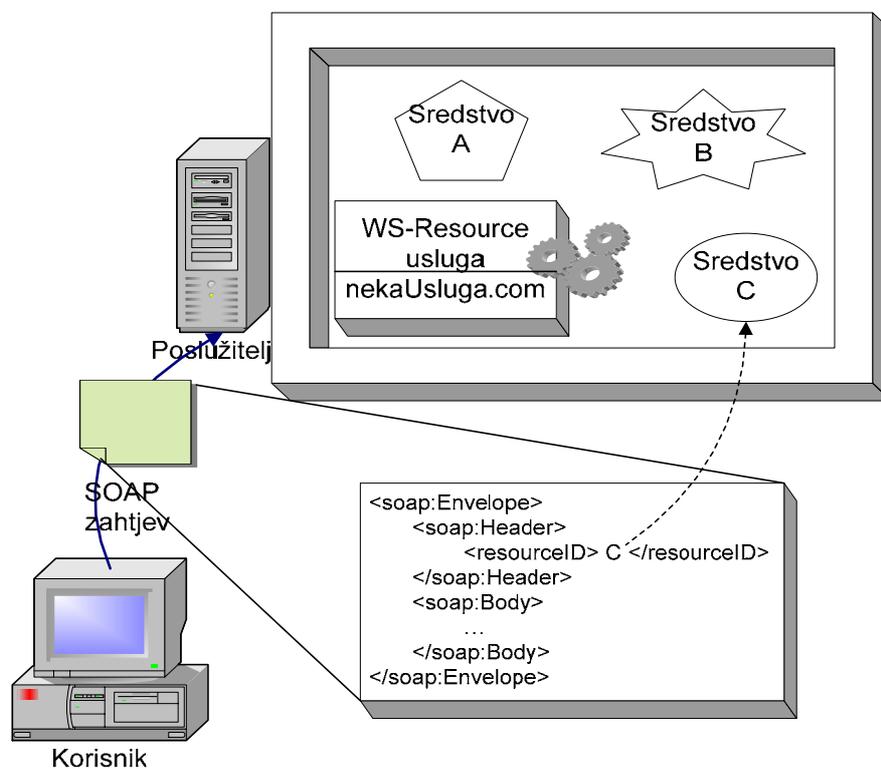
Kako bi koristio WS-Resources uslugu, korisnik mora prvo zatražiti stvaranje sredstva usluge (engl. *factory*) u kojem će se pamtili njegovo stanje izvođenja. Iznimno, u nekim slučajevima moguće je koristiti neko već postojeće sredstvo usluge. Kao rezultat stvaranja novog ili pronalaženja odgovarajućeg već postojećeg sredstva usluge korisnik dobiva referencu (engl. *endpoint reference*). Referenca je XML dokument kojim su jedinstveno definirani sredstvo i odgovarajuća usluga. Sadržaj i struktura reference specificirani su standardom WS-Addressing.



Slika 18. Stvaranje sredstava usluge i dohvat reference.

Na slici 18 prikazan je proces stvaranja sredstva usluge, te dohvaćanja odgovarajuće reference. Korisnik šalje zahtjev za stvaranjem sredstva usluge (1), poslužitelj stvara sredstvo usluge simbolički nazvano „C“ (2), te kao odgovor šalje referencu (3). Referenca sadrži adresu usluge, te informacije koje omogućavaju jedinstvenu identifikaciju odgovarajućeg sredstva usluge (engl. *reference properties*). Osim navedenog, referenca može sadržavati i druge podatke. Svi podatci osim adrese usluge su neobavezni.

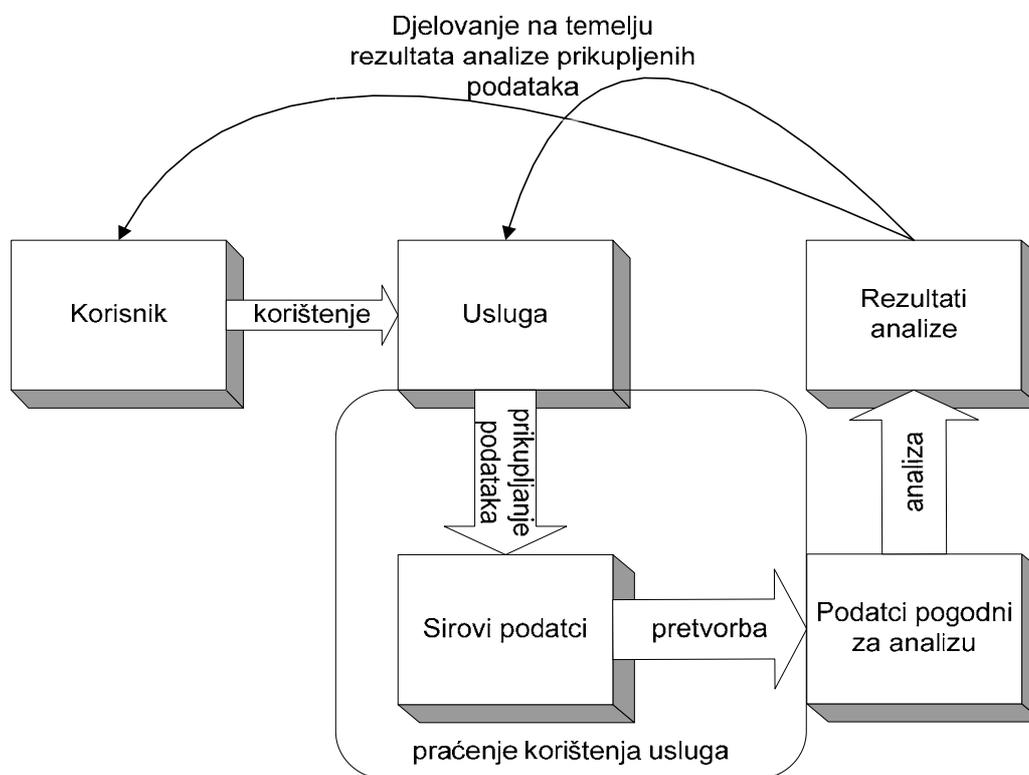
Korištenjem podataka iz reference korisnik može pristupiti usluzi (slika 19) slanjem zahtjeva na adresu usluge. Zahtjev se šalje u obliku SOAP poruke. Pritom se kao adresa usluge koristi podatak iz reference. U zaglavlju SOAP poruke specificira se da se pri izvođenju koristi sredstvo usluge „C“.



Slika 19. Pristup sredstvu WS-Resources usluge.

3. Praćenje korištenja usluga

Usluge na Internetu izvršavaju se automatski, bez neposrednog nadzora. Kako bi se otkrili eventualni problemi u radu usluge, te se pronašla rješenja kojima bi se oni otklonili, javlja se potreba za prikupljanjem informacija vezanim uz izvršavanje usluge. Zbog toga se u mnoge usluge, osim njihove osnovne logike, ugrađuje funkcionalnost kojom se prate zbivanja na poslužitelju. Praćenjem zbivanja na poslužitelju ujedno se mogu dobiti podatci o korištenju usluga koje se na njemu izvršavaju. Podatci o korištenju usluga koriste se za naplatu, te za izradu korisničkih profila. Izrada korisničkih profila je proces kojim se otkrivaju uzorci ponašanja korisnika što omogućuje prilagođavanje usluga njihovim potrebama.



Slika 20. Proces praćenja korištenja usluga.

Proces praćenja korištenja usluge prikazan je na slici 20. Za vrijeme izvođenja usluge prikupljaju se podatci o izvođenju. Prikupljeni podatci zatim se analiziraju. Na osnovu analize moguće je otkriti eventualne probleme u radu usluge, te pronaći rješenja kojim bi se oni otklonili, naplatiti korištenje usluge ili stvoriti korisnički profil.

Praćenje korištenja usluge sastoji se od dva procesa – procesa prikupljanja podataka i procesa pretvorbe sirovih podataka u oblik pogodan za analizu. Ovisno o razlozima praćenja prikupljeni podatci analiziraju se na različite načine. U ovisnosti o vrsti analize koja će se primjenjivati na prikupljenim podacima, korištenje usluga prati se različitim metodama. U nastavku poglavlja opisane su metode kojima se prati korištenje usluga, te je iznijet pregled najčešćih područja primjene praćenja korištenja usluga. Objašnjeno je koja se od opisanih metoda koristi u kojem slučaju, te su izneseni razlozi izbora pojedine metode.

3.1. Metode praćenja korištenja usluga

U ovom poglavlju objašnjene su dvije osnovne metode praćenja korištenja usluga – metoda zasebnog prikupljanja i pretvorbe podataka (engl. *batch processing*), te metoda istovremenog prikupljanja i pretvorbe podataka (engl. *streaming*) [17]. U praksi se, osim navedenih osnovnih metoda, koriste i mnoge druge metode nastale kao njihova kombinacija. Takve metode imaju svojstva obje osnovne metode. U ovisnosti o specifičnim potrebama praćenja kombiniraju se određena svojstva osnovnih metoda kako bi se postigla maksimalna učinkovitost sustava za praćenje. Bez obira na metodu koja se koristi, podatci o korištenju usluge prikupljaju se iz korisničkih zahtjeva, te iz odgovora poslužitelja.

3.1.1. Metoda zasebnog prikupljanja i pretvorbe podataka

Praćenje korištenja usluga metodom zasebnog prikupljanja i pretvorbe podataka obavlja se u dva zasebna koraka. U prvom koraku prikupljaju se podatci na način da se informacije o svim pristiglim zahtjevima i svim poslanim odgovorima zapišu u datoteku dnevnika (engl. *log file*). Zatim se, u zasebnom koraku, prikupljeni podatci pretvaraju u oblik pogodan za analizu. Slika 21 prikazuje odsječak datoteke dnevnika Web poslužitelja Apache.

```
127.0.0.1 - - [20/Apr/2004:00:11:10 +0200] "POST /potvrden_unos.htm HTTP/1.1" 200 245
127.0.0.1 - - [20/Apr/2004:00:12:17 +0200] "POST /potvrden_unos.htm HTTP/1.1" 200 198
127.0.0.1 - - [20/Apr/2004:00:13:08 +0200] "POST /potvrden_unos.htm HTTP/1.1" 200 198
127.0.0.1 - - [20/Apr/2004:00:14:30 +0200] "GET /index.htm HTTP/1.1" 200 488
127.0.0.1 - - [20/Apr/2004:00:14:30 +0200] "GET /welcome.htm HTTP/1.1" 200 135
127.0.0.1 - - [20/Apr/2004:00:14:30 +0200] "GET /izbornik.htm HTTP/1.1" 200 5365
```

Slika 21. Odsječak datoteke dnevnika Web poslužitelja Apache

Budući da su proces prikupljanja podataka i pretvorbe odvojeni, za vrijeme izvođenja usluge obavlja se samo prikupljanje podataka. Prednost ove metode je što je usporavanje izvođenja usluge radi praćenja njenog korištenja zanemarivo. Budući da se prikupljaju podatci o svim zahtjevima i odgovorima, prikupljeni podatci vrlo su općeniti, te je moguće jednom prikupljene podatke analizirati na više proizvoljnih načina. Kod praćenja korištenja usluga metodom zasebnog prikupljanja i pretvorbe podataka proces prikupljanja podataka izuzetno je jednostavan. Za razliku od toga, proces pretvorbe može biti vrlo zahtjevan. Zbog toga je ova metoda praćenja pogodna u situacijama kada za analizu nije potrebna neka složena struktura podataka. Nedostak ovakve metode u kojoj se spremaju podatci o svim zahtjevima i odgovorima je činjenica da količina spremljenih podataka brzo raste. Velike količine podataka dodatno otežavaju proces pretvorbe.

3.1.2. Metoda istovremenog prikupljanja i pretvorbe podataka

Za razliku od prethodno opisane metode, kod ove metode praćenja korištenja usluga, proces prikupljanja podataka i proces pretvorbe prikupljenih podataka u oblik pogodan za analizu odvijaju se istovremeno. Na kraju izvođenja usluge, a prije slanja odgovora korisniku, poslužiteljska logika izdvaja potrebne dijelove korisničkog zahtjeva i generiranog odgovora. Izdvojeni podatci povezuju se s podacima dobivenim iz prethodnih zahtjeva u oblik pogodan za analizu, te se u takvom obliku spremaju. Za spremanje podataka o korištenju usluge uglavnom se koristi relacijska baza podataka. Relacijska baza podataka omogućava napredne metode povezivanja podataka, te je zbog toga pogodna za korištenje u sklopu ove metode praćenja korištenja usluga.

Prednost metode istovremenog prikupljanja i pretvorbe podatka je što su prikupljeni podatci odmah spremni za analizu. Također, podatci se prikupljaju selektivno što sprečava njihovo nagomilavanje. Međutim, da navedene prednosti dođu do izražaja potrebno je unaprijed poznavati kakva analiza će se vršiti nad prikupljenim podacima. Zbog toga oni gube na općenitosti. Obrada podataka u trenutku njihova prikupljanja jednostavnija je od obrade prethodno prikupljenih općenitih podataka. Zbog toga je upotreba ove metode pogodna u situacijama kada je, za potrebe analize, podatke potrebno grupirati u složene strukture. Budući da se pretvorba podataka obavlja tijekom izvođenja usluge, praćenje korištenja usluga ovom metodom usporava izvođenje usluge u većoj mjeri nego što je to slučaj kod metode zasebnog prikupljanja i pretvorbe podataka.

3.2. Područja primjene praćenja korištenja usluga

3.2.1. Praćenje korištenja usluga radi osiguravanja ispravnog rada

Djelovanje mnogih organizacija, te poslovanje mnogih tvrtki zasniva se na Internet uslugama. Budući da usluge na Internetu dobivaju sve veći značaj, važno je osigurati ispravno izvršavanje usluga, te omogućiti nesmetan pristup korisnika usluzi u svakom trenutku. Problemi u radu usluge mogu se javiti zbog pogreške u izvođenju usluge, prevelikog opterećenja poslužitelja usluge, te zbog napada na poslužitelj usluge. Navedeni problemi javljaju se zbog propusta u razvoju usluge ili poslužitelja. Brzi razvoj novih tehnologija, te pritisak za što bržim izlaskom na tržište doprinose tome da na tržište izlaze nedovoljno testirani proizvodi. Budući da je sigurnosne propuste najteže otkriti, upravo su oni najčešći uzrok problema u radu usluge.

Ciljevi napada na poslužitelje su onemogućavanje pristupa poslužitelju (engl. *Denial of Service*), promjena sadržaja usluge i slično. Bez obzira na razloge i metode napada na poslužitelje, oni donose velike štete pružateljima usluga. U pokušaju njihovog sprečavanja koriste se dvije metode. Jedna metoda sprečavanja napada je njihovo onemogućavanje uklanjanjem sigurnosnih propusta sa poslužitelja. Druga metoda je pokušaj obeshrabrivanja potencijalnih napadača prijeteći im kaznenim progonom. Pritom je dodatni problem činjenica da napad može biti izvršen sa udaljenog računala, koje se može nalaziti u drugoj državi, pa osim tehnoloških

problema identifikacije napadača, postoje i pravne prepreke koje otežavaju njihovo kažnjavanje.

Budući da proizvođači poslužitelja uklanjaju otkrivene sigurnosne propuste, napadi su mogući jedino u slučajevima kada zlonamjerni korisnici prvi otkriju sigurnosni propust poslužitelja. Kako bi se spriječili višestruki napadi korištenjem istog sigurnosnog propusta, želja je proizvođača otkriti sigurnosni propust korišten za napad. Otkrivanje sigurnosnog propusta moguće je analizom slijeda zbivanja na poslužitelju za vrijeme trajanja napada. Identifikacija napadača, nužna za metodu obeshrabrivanja prijetnjom kaznenim progonom, zasniva se također na analizi podataka prikupljenih za vrijeme napada na poslužitelj. U ovom slučaju, umjesto slijeda zbivanja, analizira se izvor zlonamjernih zahtjeva.

Za potrebe analize podataka prikupljenih za vrijeme napada na poslužitelj nije potrebno povezivati podatke u složene strukture. Zbog toga se za praćenje koristi metoda zasebnog prikupljanja i pretvorbe podataka. Također, analiziraju se isključivo podatci prikupljeni za vrijeme napada. Budući da je kriterij za izdvajanje podataka potrebnih za analizu vrijeme, lako je izdvojiti ih iz gomile. Upotreba metode istovremenog prikupljanja i pretvorbe podataka nepotrebno bi usporavala izvršavanje usluge.

3.2.2. Praćenje korištenja usluga radi naplate

U razdoblju prije nastanka Weba, Internet usluge bile su razvijane uglavnom u akademske i istraživačke svrhe. Razvojem Weba mijenja se njihova namjena. Danas se na Internetu nudi veliki broj komercijalnih usluga. Razvojem komercijalnih usluga javlja se potreba za praćenjem njihova korištenja radi naplate.

Primjer komercijalne usluge je Web trgovina. Za uslugu prodaje ključna je mogućnost zabilježavanja podataka o kupljenoj robi. Ti se podatci zatim koriste za dostavu i naplatu. Prikupljanje podataka o naručenoj robi jedan je od oblika praćenja korištenja usluge. Iz zahtjeva poslanih Web poslužitelju potrebno je izdvojiti one kojima se naručuje proizvod. Te je zahtjeve zatim potrebno povezati s naručiteljem čiji je

identitet ustanovljen na osnovu nekog ranijeg zahtjeva. Mogućnost odustajanja od narudžbe dodatno otežava taj zadatak. Slično vrijedi i za druge Web usluge.

U ovom slučaju praćenje metoda zasebnog prikupljanja i pretvorbe podataka nije prikladno, budući da bi pronalaženje bitnih informacija i njihovo povezivanje filtriranjem datoteke dnevnika bilo izuzetno složeno. Zbog toga se u ovom slučaju za praćenje koristi metoda istovremenog prikupljanja i pretvorbe podataka. Umjesto da se zabilježavaju svi zahtjevi, zabilježavaju se samo oni kojima se nešto naručuje. To se čini tako da po primitku zahtjeva kojim se nešto naručuje poslužiteljska logika informaciju o narudžbi sprema u bazu podataka istovremeno je povezujući sa podacima o identitetu naručitelja.

3.2.3. Praćenje korištenja usluga radi izrade korisničkih profila

Usporedo razvoju računalnih mreža, zahvaljujući napretku u računalnoj tehnologiji, povećavaju se brzine rada procesora i kapaciteti uređaja za spremanje podataka. Takav napredak učinio je dostupnima računala pogodna za učinkovitu obradu velikih količina podataka. Metode koje su do prije nekoliko godina bile moguće samo obavještajnoj zajednici (engl. *intelligence*) pomoću izuzetno skupih super-računala (engl. *mainframe computers*), danas koristi gotovo svaki trgovački lanac. Budući da se metode obavještajne zajednice koriste u poslovne svrhe, ovo novo područje uporabe računala naziva se poslovno zaključivanje (engl. *Business Intelligence*). Radi se o sprezi između računalnih i ekonomskih znanosti, s ciljem da se računalnom obradom velikih količina podataka otkriju uzorci ponašanja potrošača, kako bi se olakšalo donošenje odluka.

Tipični primjer upotrebe metoda poslovnog zaključivanja su trgovački lanci. Podatci se prikupljaju na blagajnama. Kako bi identificirali kupce trgovački lanci izdaju posebne identifikacijske kartice, a kupci se na njihovo korištenje motiviraju posebnim uvjetima kupovine (npr. popusti, nagradne igre i slično). Obradom prikupljenih podataka podataka stvaraju se uvjeti za lakše donošenje ispravnih odluka, uglavnom vezanih za marketing.

Premještanjem trgovina na Web moguće je prikupiti mnogo više podataka o ponašanju potrošača. Dok je u „običnim“ trgovinama jedino mjesto prikupljanja podataka blagajna, na Webu je ponašanje potrošača moguće pratiti mnogo detaljnije. U Web trgovini svaki zahtjev korisnika nosi neku korisnu informaciju, bilo da se radi o zahtjevu za pregledom detalja o nekom proizvodu, uvidom u cijenu, odustajanju od kupnje ili nešto slično. Povezivanjem takvih informacija moguće je dobiti detaljan uvid u razloge nekog zamijećenog obrasca ponašanja, te time stvoriti uvjete za donošenje odluka kojima bi se poboljšala prodaja. Na primjer, moguće je primijetiti obrazac odustajanja od kupovine nakon uvida u cijenu ili nešto slično. Osim toga, na Webu se može pratiti i učinkovitosti svake pojedine reklame (engl. *banner*), tako da se prati broj dolazaka u Web trgovinu putem svake od njih. Osim na Web trgovine, opisani pristup analizi podataka prikupljenih praćenjem korištenja usluga odnosi se i na ostale Web usluge.

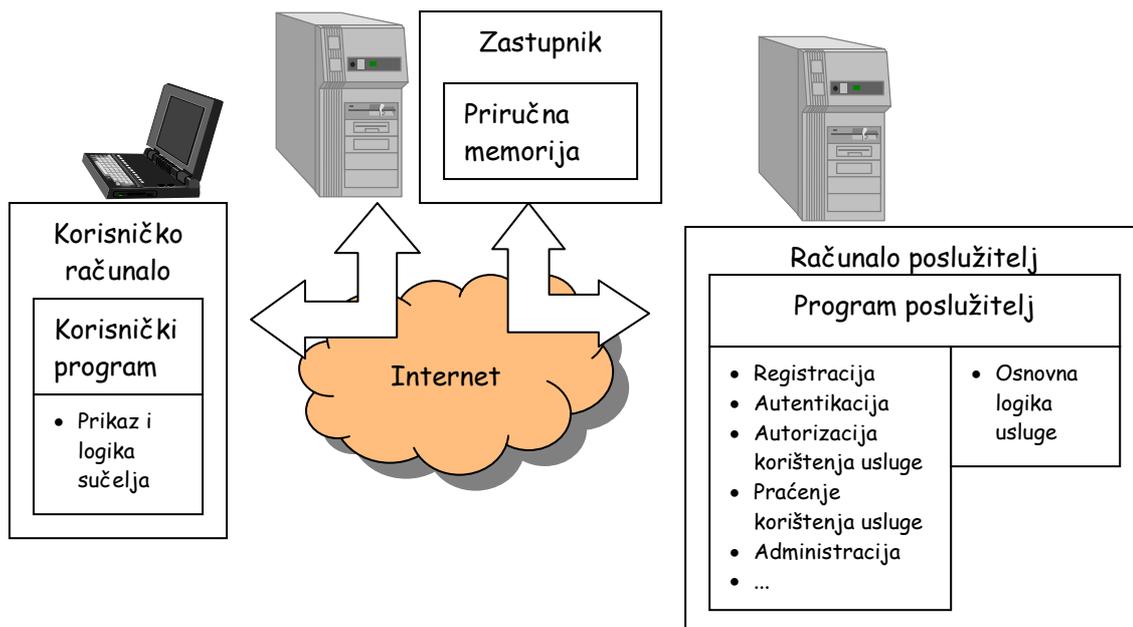
Za potrebe izrade korisničkih profila obje opisane metode praćenja korištenja usluga pokazuju neke dobre i neke loše osobine. Dobra osobina metode zasebnog prikupljanja i pretvorbe podataka je što njenim korištenjem prikupljeni podatci zadržavaju općenitost. Jednom prikupljene podatke moguće je zatim analizirati na više proizvoljnih načina, što omogućuje veću fleksibilnost pri izradi korisničkih profila. Loša osobina je izuzetno složeno povezivanje prikupljenih podataka za potrebe pretvorbe u oblik pogodan za analizu. Metoda istovremenog prikupljanja i pretvorbe podataka rješava problem povezivanja podataka, ali usporava rad poslužitelja jer se po primitku svakog zahtjeva mora izvršiti logika koja pretvara prikupljene podatke u oblik pogodan za analizu. Iz navedenih razloga u praksi se koriste obje navedene metode ili neka njihova kombinacija.

4. Posrednički sustav MidArc

Od nastanka prvih usluga, proces razvoja nove usluge nastojao se je pojednostavniti. Nakon što je primjećeno da se pri razvoju svake nove usluge ispočetka razvija i modul za komunikaciju, te da su moduli za komunikaciju različitih usluga međusobno slični, mnogo je truda uloženo u razvoj standardne komunikacijske infrastrukture. Tako se danas, pri razvoju nove usluge programeri više ne trebaju brinuti o komunikaciji, već se mogu potpuno posvetiti razvoju logike usluge. Također, postoje razvojna okruženja koja olakšavaju razvoj usluga nudeći gotove module za neke često korištene funkcionalnosti. Primjer takvog okruženja je ASP.NET u sklopu kojeg se nudi mnoštvo gotovih modula uglavnom namijenjenih izradi grafičkog sučelja (npr. kalendar).

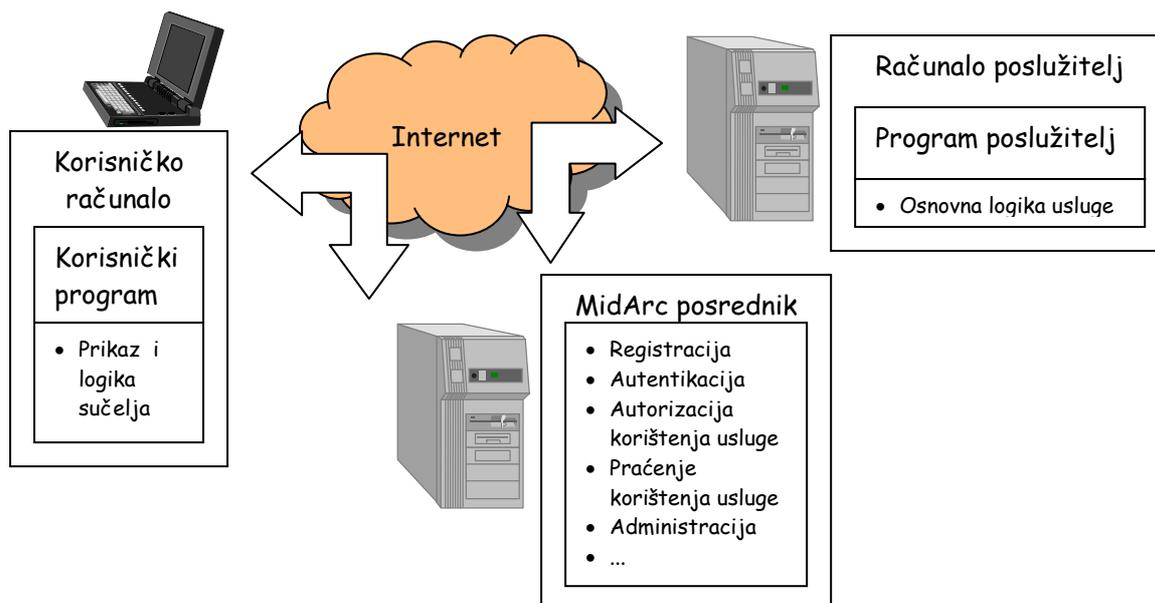
U međuvremenu, razvojem novih usluga, javlja se potreba za praćenjem njihova korištenja, provođenjem poslovne politike, izvođenjem usluge u sigurnom okruženju i slično. Kako bi se zadovoljile navedene potrebe, velik broj usluga, osim same logike usluge, ostvaruje i dodatne funkcionalnosti. Među njih ubrajamo registraciju i autentikaciju korisnika, autorizaciju i praćenje korištenja usluge i druge slične funkcionalnosti. Navedene funkcionalnosti zajedničke su velikom broju usluga, te se zbog toga nazivaju zajedničkim funkcionalnostima. Budući da ne ovise o osnovnoj logici usluge, zajedničke funkcionalnosti ostvarene su na gotovo isti način u sklopu međusobno različitih usluga. Iz navedenog razloga moguće ih je izdvojiti iz pojedinih usluga, te objediniti i ostvariti kao dio zasebnog sustava zajedničkih funkcija. Time se razvoj usluga dodatno pojednostavljuje jer programeri više ne moraju razvijati navedene funkcionalnosti zasebno za svaku uslugu. Umjesto toga, usluge se u svom radu oslanjaju na sustav zajedničkih funkcija.

Sustav zajedničkih funkcija, zbog specifičnosti funkcija koje obavlja, mora se nalaziti u komunikacijskom kanalu između korisnika i usluga. Budući da posreduje u njihovoj komunikaciji, takav je sustav nazvan posrednikom. Primjer posredničkog sustava je MidArc posrednik koji se razvija u suradnji Zavoda za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu i tvrtke Ericsson Nikola Tesla.



Slika 22. Komunikacija posredovanjem zastupnika.

Mjesto u komunikaciji između korisnika i usluga, namijenjeno posredniku, sada zauzimaju zastupnici (engl. *proxy*) (slika 22). Uloga zastupnika je rasterećenje poslužitelja spremanjem podataka s poslužitelja u priručnu memoriju. U slučaju da je više istih zahtjeva upućeno poslužitelju posredstvom istog zastupnika, neki podatci mogu se pronaći na zastupniku. Kao rezultat, tada nema potrebe za slanjem zahtjeva do poslužitelja, već se traženi podatci dohvaćaju sa zastupnika čime se rasterećuje poslužitelj. Međutim, zastupnici mogu spremati isključivo statičke podatke (npr. slike), dok sve funkcionalnosti i dalje ostaju na poslužitelju. Posrednik donosi značajnije rasterećenje poslužitelja obavljajući dio poslužiteljskih funkcionalnosti (slika 23). S obzirom na navedeno, posrednika možemo smatrati oblikom unaprijeđenog zastupnika.



Slika 23. Komunikacija posredovanjem MidArc posrednika.

4.1. Funkcionalnosti MidArc posrednika

U trenutnoj inačici, MidArc posrednik ostvaruje sljedeće zajedničke funkcionalnosti: registraciju, autentikaciju, autorizaciju pristupa usluzi, bilježenje korištenja usluge, identifikaciju, pretraživanje korisničkih podataka i podataka usluga, te administraciju.

Kako bi mogli komunicirati posredstvom MidArc posrednika usluge i korisnici moraju se prethodno registrirati. Prilikom registracije usluge MidArc posrednik prikuplja podatke o korisnicima i uslugama nužne za ispravan rad posrednika. Za usluge se prikupljaju podatci kao što su: URL usluge, opis usluge, uvjete korištenja usluge, pravila za praćenje korištenja usluge i drugo. Od korisnika se traže ime i prezime, datum rođenja, adresa i slično. Registriranim korisnicima i uslugama dodjeljuju se korisničko ime i zaporka. Njih korisnici i usluge koriste za daljnju komunikaciju s posrednikom. Registracija se obavlja samo jednom, tijekom prvog posjeta korisnika ili administratora usluge posredniku. Nakon što su prikupljeni, podatci se spremaju na posredniku, te se više ne traže. Podatke prikupljene za vrijeme registracije korisnici i administratori usluga mogu naknadno promijeniti.

Autentikacija je proces utvrđivanja identiteta. Na MidArc posredniku korisnici i usluge autenticiraju se prilikom svake prijave za rad sa sustavom. Prilikom prijave, korisnici i usluge predstavljaju se korisničkim imenom, a svoj identitet dokazuju lozinkom. Na osnovu upisanog korisničkog imena i lozinke MidArc posrednik utvrđuje identitet korisnika ili usluge. Uspješnom autentikacijom korisnika uspostavlja se korisnička sjednica, dok se autentikacijom poslužitelja usluge uspostavlja poslužiteljska sjednica. Sjednica traje do odjave sa sustava. Poslužiteljske sjednice obično traju duže nego korisničke, pa se usluge rjeđe autenticiraju.

Autorizacija je proces provjere prava pristupa uslugama. Različiti korisnici imaju različita prava pristupa uslugama. Pravo pristupa korisnika uslugama ovisi o mnogo uvjeta. Na primjer, usluga može nuditi različite pretplatničke pakete. Pravo korisnika na pristup takvoj usluzi ovisi o paketu na koji se korisnik pretplatio. MidArc posrednik, prilikom svakog korisničkog zahtjeva, provjerava ima li korisnik pravo na zatraženu uslugu. Provjera se izvodi pregledom podataka o pravima registriranih korisnika, koji su spremljeni na MidArc sustavu. Ako se ustanovi da poziv usluge nije u skladu s pravima pozivatelja akcija se ne odobrava.

Bilježenje korištenja usluga je proces promatranja i bilježenja poziva usluga upućenih putem MidArc posrednika. Bilježenje korištenja usluga provodi se na osnovu uvida u zahtjeve upućene uslugama putem MidArc posrednika, te njihove odgovore. Prikupljene podatke mogu pregledavati i korisnici i usluge. Svatko može pregledavati podatke o zahtjevima koje je uputio, dok usluge mogu dodatno pregledavati podatke o primljenim zahtjevima.

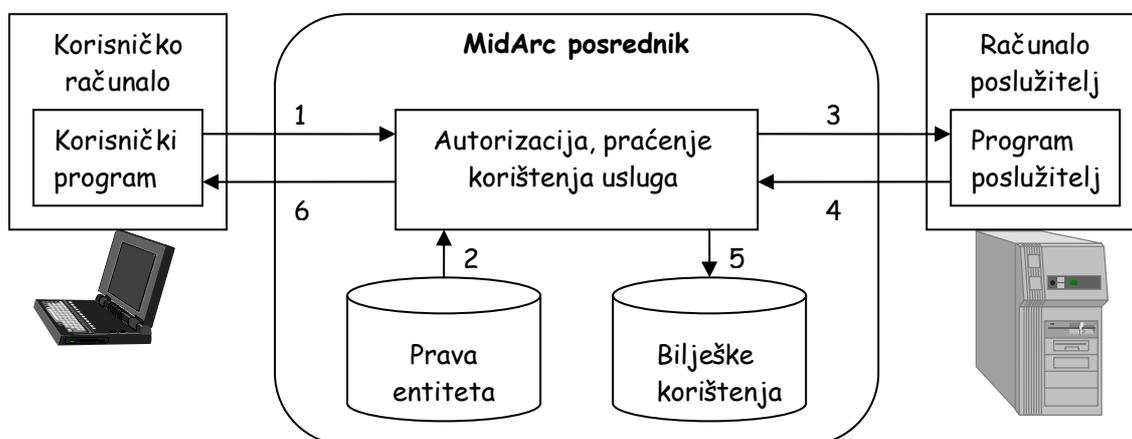
MidArc posredniku poznat je identitet svakog korisnika prijavljenog za rad sa sustavom. Utvrđivanje identiteta obavljeno je tijekom prijave, procesom autentikacije. Budući da se MidArc posrednik brine o autorizaciji pristupa i praćenju korištenja usluga, uslugama često nije bitan identitet pozivatelja. Međutim, postoje situacije u kojima usluzi može biti bitan identitet pozivatelja. Na primjer, u slučaju kada se sadržaj usluge ili izgled korisničkog sučelja prilagođava korisniku. U takvom slučaju, usluga može zatražiti podatke o identitetu korisniku od MidArc posrednika. Takav proces naziva se identifikacija. Osim identiteta, procesom identifikacije usluga može dohvatiti i druge podatke o korisniku. Dostupnost podataka o korisniku ovise o tome

koji su podatci o njemu prikupljeni tijekom registracije, te o pravima usluge za pristup navedenim podacima.

Funkcija administracije namijenjena je administratoru MidArc posrednika. Zadatci administratora MidArc posrednika su uređivanje prava pristupa korisnika i usluga, promjena postojećih stavaka u skladu s potrebama, svrstavanje korisnika i usluga u grupe i drugo.

4.2. Način rada MidArc posrednika

Funkcionalnosti MidArc posrednika, opisane u prethodnom poglavlju, izvršavaju se različitim učestalošću. Tako se, na primjer, funkcija registracija obavlja relativno rijetko, po jednom za svaku novu uslugu ili korisnika. Funkcija autentikacije izvršava se po jednom za svaku prijavu za rad sa sustavom. Funkcije autorizacije i praćena korištenja usluga izvršavaju se najčešće, prilikom svakog poziva usluge.



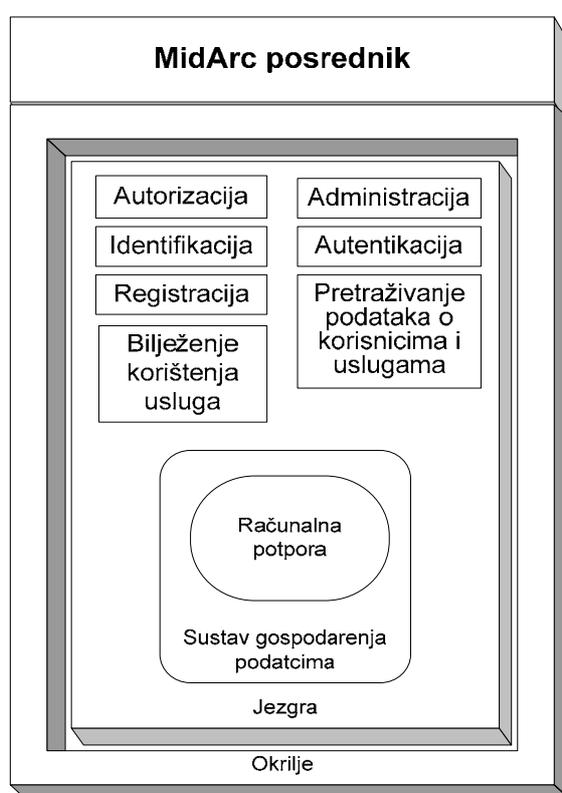
Slika 24. Način rada MidArc posrednika prilikom poziva usluge.

Na slici 24 prikazan je način rada MidArc posrednika prilikom poziva usluge. Zahtjev namijenjen usluzi korisnik šalje MidArc posredniku (1), posrednik autorizira zahtjev (2), te ga prosljeđuje poslužitelju usluge (3). Nakon izvršavanja, usluga šalje odgovor

MidArc posredniku (4), koji bilježi korištenje usluge (5) i prosljeđuje odgovor korisničkom programu (6).

4.3. Arhitektura MidArc posrednika

Arhitektura MidArc posrednika je slojevita. Može se podijeliti na dvije osnovne cjeline – jezgru MidArc posrednika i okrilje MidArc posrednika. Arhitektura MidArc posrednika prikazana je na slici 25.



Slika 25. Programska arhitektura MidArc posrednika.

Okrilje MidArc posrednika čini mehanizam za pristup jezgri. Okrilje omogućava korisničkim programima i programima poslužiteljima upotrebu jezgrinih funkcionalnosti. Dodatno, okrilje omogućava ostvarivanje komunikacije između korisnika i poslužitelja usluga, te komunikaciju među sustavima ostvarenima korištenjem različitih tehnologija.

Jezgra MidArc posrednika sadrži ostvarenja zajedničkih funkcionalnosti, te potporu njihovu izvođenju. Potporu izvođenju zajedničkih funkcionalnosti pruža sustav

gospodarenja podacima. Sustav gospodarenja podacima upravlja podacima koristeći računalnu potporu. Računalna potpora obuhvaća organizaciju spremljenih podataka i skup funkcija kojima se omogućava brzi dohvat podataka, te učinkovito upravljanje podacima. Zajedničke funkcionalnosti ostvarene u jezgri MidArc posrednika su: autentikacija, autorizacija pristupa usluzi, identifikacija, pretraživanje korisničkih podataka i podataka usluga, administracija, registracija, te bilježenje korištenja usluge. Funkcionalnosti registracije i bilježenja korištenja usluge zajedno čine podsustav za praćenje korištenja usluga. Podsustav za praćenje korištenja usluga detaljnije je opisan u sljedećem poglavlju.

5. Organizacija i nadogradnja podsustava za praćenje korištenja usluga

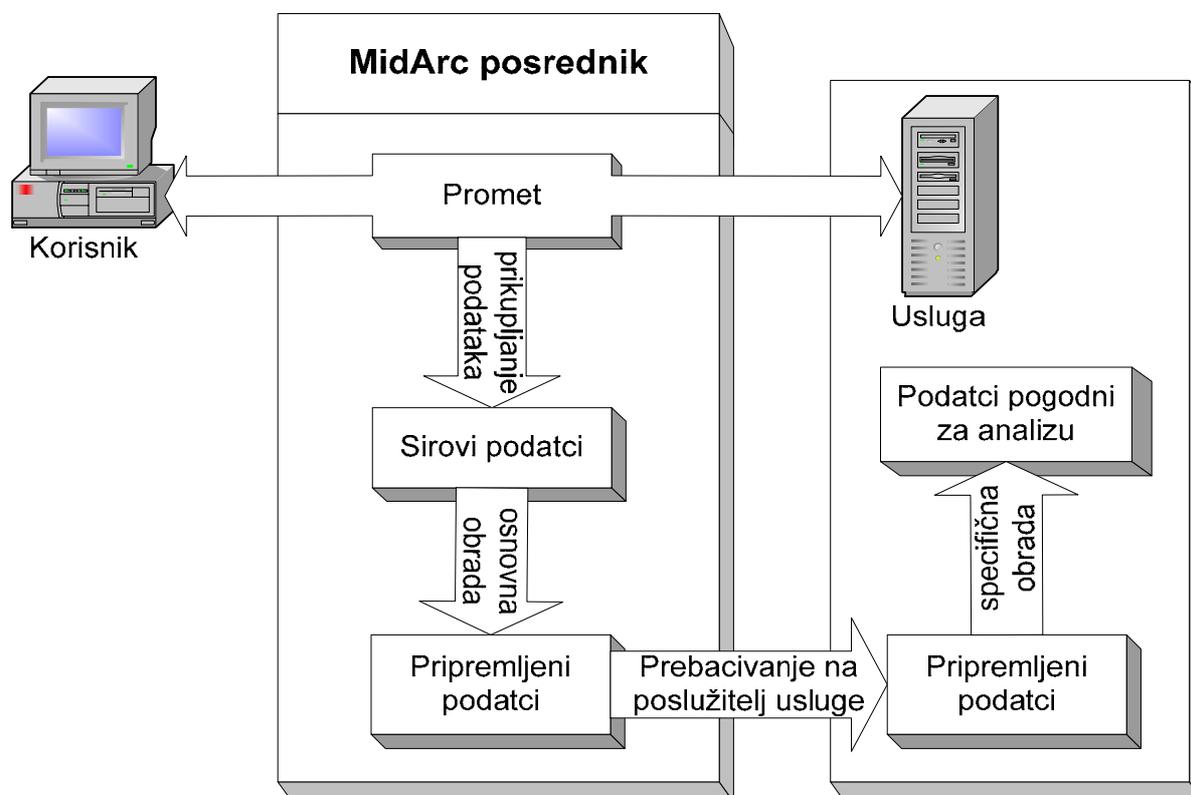
Funkcionalnost praćenja korištenja usluga putem MidArc posrednika razvijena je za praćenje korištenja Web Services usluga. U međuvremenu, razvijen je novi standard pružanja usluga pod nazivom WS-Resources. Razvojem WS-Resources tehnologije javlja se potreba za omogućavanjem praćenja korištenja sredstava usluga putem MidArc posrednika. U sklopu praktičnog dijela rada unaprijeđen je podsustav za praćenje korištenja Web Services usluga putem MidArc posrednika kako bi se omogućilo praćenje WS-Resources usluga. Razvijen je modul za bilježenje korištenja sredstava usluga. Dodatno, kao pomoćni modul, nadograđen je modul za registraciju usluga. Budući da WS-Resources standard predstavlja proširenje Web Services standarda, razvijeni moduli mogu se koristiti i za praćenje korištenja Web Services usluga.

U nastavku poglavlja opisana je metoda koja se koristi za praćenje korištenja usluga putem MidArc posrednika. Objašnjen je razlog izbora navedene metode, te odnos s metodama praćenja korištenja usluga opisanim u poglavlju 3.1. Prikazana je arhitektura ostvarenja podsustava za praćenje korištenja usluga na MidArc posredniku. Zatim je opisana arhitektura modula za registraciju usluga na MidArc posredniku, s posebnim naglaskom na modulu za zadavanje pravila praćenja korištenja usluga i njihovih sredstava. Na kraju je opisan modul za praćenje korištenja sredstava usluga.

5.1. Analiza metode praćenja korištenja usluga u sklopu MidArc posrednika

MidArc posrednik izvodi funkcionalnost praćenja korištenja usluga za više međusobno različitih usluga od kojih svaka ima svoje specifičnosti. Osim razlike u sadržaju samih usluga, razlikuju se i razlozi praćenja rada pojedinih usluga. U skladu s time, razlikuju se i metode analize prikupljenih podataka. Različite metode analize zahtijevaju različito pripremljene podatke. Zbog velikih razlika u navedenim zahtjevima, u sklopu MidArc posrednika ne izvodi se proces pretvorbe prikupljenih

podataka u oblik pogodan za analizu. Pretvorba podataka prikupljenih praćenjem korištenja usluge putem MidArc posrednika u oblik pogodan za analizu prepušta se usluzi. Metoda istovremenog prikupljanja i pretvorbe podataka nije pogodna za korištenje u slučajevima u kojima se pojedini koraci izvode na različitim sustavima, te zbog toga u sklopu MidArc posrednika praćenje korištenja usluga nije ostvareno navedenom metodom. Metoda zasebnog prikupljanja i pretvorbe podataka pogodna je za korištenje u slučajevima u kojima se pojedini koraci izvode na različitim sustavima. Međutim, zbog izuzetne složenosti procesa pretvorbe podataka prikupljenih navedenom metodom, ona nije pogodna za slučajeve u kojima su za analizu potrebne složene strukture podataka. Tako, navedena metoda nije pogodna za praćenje korištenja usluga radi naplate. Budući da je naplata korištenja usluga jedan od razloga njihovog praćenja putem MidArc posrednika, praćenje korištenja usluga u sklopu MidArc posrednika nije ostvareno metodom zasebnog prikupljanja i pretvorbe podataka.



Slika 26. Praćenje korištenja usluga na MidArc posredniku.

U sklopu MidArc posrednika, za praćenje korištenja usluga koristi se kombinacija navedenih osnovnih metoda praćenja korištenja usluga. Proces pretvorbe prikupljenih podataka u oblik pogodan za analizu podijeljen je na dva koraka. Na MidArc posredniku izvodi se proces prikupljanja podataka, te prvi korak procesa pretvorbe podataka u oblik pogodan za analizu, dok se drugi korak prepušta usluzi (slika 26).

U prvom koraku, nad prikupljenim podacima provode se osnovni postupci obrade. Osnovnom obradom podatci se pripremaju za nastavak obrade. Pritom je važno da se podatci pripreme na način da je izvođenje sljedećeg koraka pretvorbe podataka jednostavnije od izvođenja pretvorbe nad sirovim podacima. Također, navedeni pripremni postupci moraju biti dovoljno općeniti da bi ih se moglo izvesti na MidArc posredniku, odnosno bez poznavanja semantike usluge. Pripremni postupci uključuju izdvajanje podataka iz poruka korisničkog zahtjeva i poslužiteljskog odgovora, te grupiranje izdvojenih podataka s osnovnim podacima o pozivu usluge. Izdvajanje podataka izvodi se na osnovu pravila praćenja zadanih pri registraciji usluge. Izdvojeni podatci grupiraju se s osnovnim podacima o pozivu usluge, te se spremaju na MidArc posredniku, dok se ostali podatci odbacuju. Osnovni podatci o pozivu usluge uključuju identifikatore korisnika i usluge, identifikator pozvane procedure, identifikator autoriziranosti poziva usluge, te vrijeme upućivanja zahtjeva i odgovora. Proces prikupljanja podataka o korištenju usluga, proces izdvajanja podatka odabranih za praćenje i proces grupiranja izdvojenih podataka s osnovnim podacima o pozivu usluge izvode se u jednom koraku.

U drugom, završnom koraku obrade podataka o korištenju usluge, podatci se obrađuju u skladu sa semantikom usluge, razlozima njezinog praćenja, te metodom kojom će se prikupljeni podatci analizirati. Završna obrada podataka uključuje međusobno povezivanje podataka izdvojenih iz različitih zahtjeva i odgovora. Drugi korak obrade podataka o korištenju usluge izvodi se na poslužitelju usluge. Prebacivanje pripremljenih podataka o korištenju usluge sa MidArc posrednika na poslužitelj usluge izvodi se na zahtjev usluge. Trenutak izvođenja drugog koraka obrade podataka ovisi isključivo o potrebama usluge.

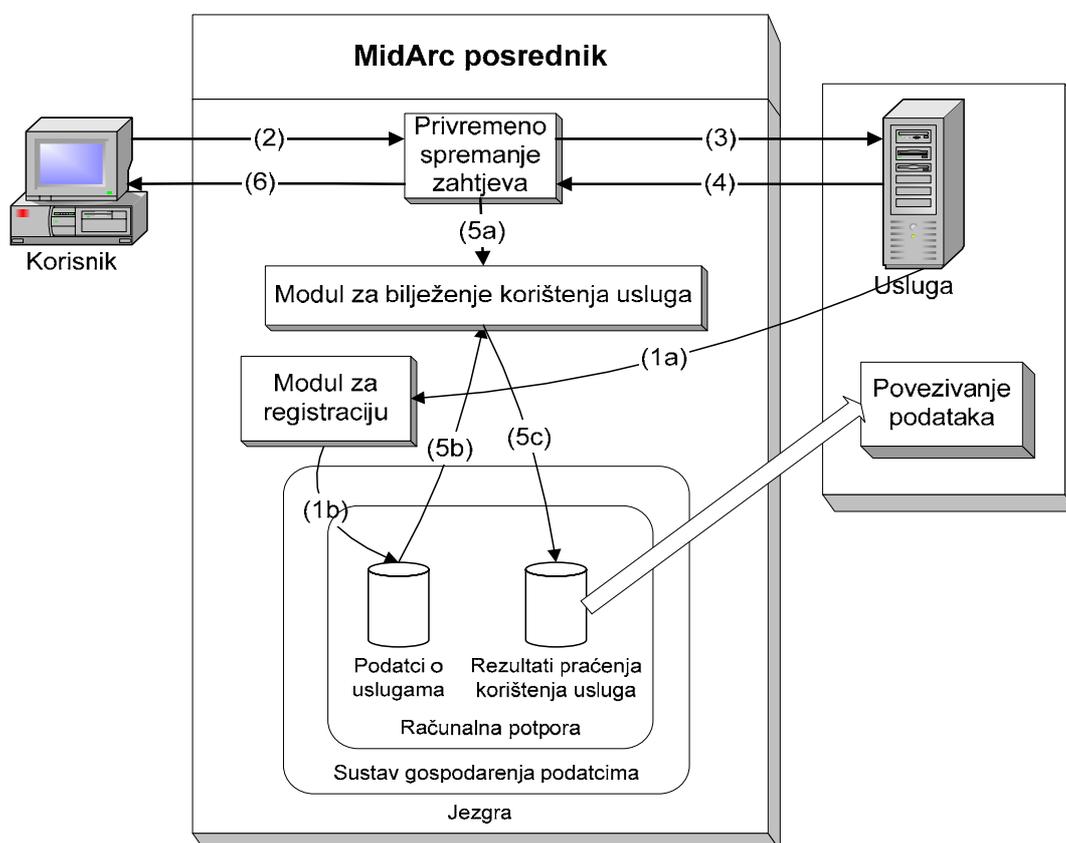
Opisana metoda praćenja korištenja usluga posjeduje prednosti obje osnovne metode praćenja korištenja usluga. Podatci se predaju na obradu usluzi u obliku pogodnom za različite metode analize, kao kod metode zasebnog prikupljanja i obrade podataka. Nedostatak metode zasebnog prikupljanja i obrade podataka je izuzetno složen proces pretvorbe podataka u oblik pogodan za analizu. Pretvorba podataka u oblik pogodan za analizu djelomično se izvodi na MidArc posredniku čime se uklanja navedeni nedostatak, te se rasterećuje usluga. Također, podatci se prikupljaju selektivno čime se štedljivije koriste kapaciteti spremnika, kao kod metode istovremenog prikupljanja i obrade podataka.

5.2. Arhitektura podsustava za praćenje korištenja usluga

Na slici 27 prikazana je arhitektura podsustava za praćenje korištenja usluga u sklopu MidArc posrednika. Podsustav za praćenje korištenja usluga sastoji se od modula za registraciju, modula za bilježenje korištenja usluga, spremnika podataka o uslugama i spremnika podataka prikupljenih praćenjem korištenja usluga. Pomoću navedenih modula MidArc posrednik omogućava prilagodbu praćenja korištenja usluga za svaku pojedinu uslugu. Prilagodba praćenja korištenja usluge provodi se zadavanjem pravila praćenja. Pravila praćenja korištenja usluge zadaje administrator usluge za vrijeme registracije usluge (1a). Pravila praćenja korištenja usluge spremaju se na MidArc posredniku zajedno s ostalim podacima o usluzi prikupljenim za vrijeme registracije usluge (1b).

Nakon uspješno završene registracije usluge, MidArc posrednik prati njezino korištenje u skladu sa zadanim pravilima praćenja. Pri svakom pozivu usluge, korisnički zahtjev upućuje se usluzi putem MidArc posrednika (2). Zahtjev se privremeno sprema na posredniku, dok se kopija zahtjeva proslijeđuje usluzi kojoj je zahtjev namijenjen (3). Po završetku izvođenja usluge, odgovor poslužitelja usluge šalje se putem MidArc posrednika (4). Funkcionalnost modula za praćenje korištenja usluga izvodi se u trenutku kada su na MidArc posredniku dostupne obje poruke – poruka korisničkog zahtjeva, te poruka odgovora poslužitelja usluge (5a). Iz spremišta podataka o uslugama dohvaćaju se pravila za praćenje korištenja pozvane usluge (5b). Na osnovu dohvaćenih pravila praćenja korištenja usluge, modul za

bilježenje korištenja usluga izdvaja podatke odabrane za praćenje iz poruka korisničkog zahtjeva i odgovora poslužitelja. Izdvojeni podatci grupiraju se s osnovnim podacima o pozivu usluge u odgovarajući zapis. Stvoreni zapis predstavlja rezultat praćenja korištenja usluge, te se sprema u bazu podataka MidArc posrednika (5c). Na kraju procesa praćenja, poruka odgovora poslužitelja prosljeđuje se korisniku (6). Podatci prikupljeni praćenjem korištenja usluge na MidArc posredniku prepuštaju se usluzi na daljnju obradu. Daljnja obrada uglavnom uključuje međusobno povezivanje podataka izdvojenih iz različitih zahtjeva i odgovora.



Slika 27. Arhitektura podsustava za praćenje korištenja usluga.

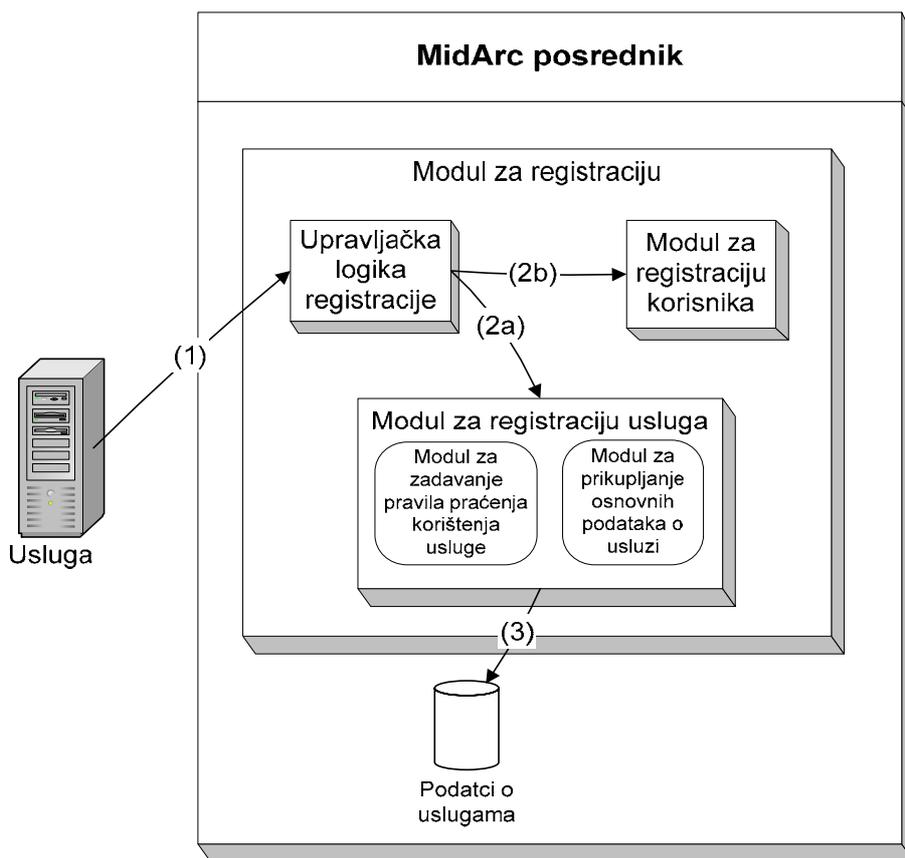
5.3. Nadogradnja modula za registraciju usluga

Na slici 28 prikazana je organizacija registracije u MidArc posredniku. Ostvarenje modula za registraciju [19] smješteno je u jezgri MidArc posrednika, zajedno s ostvarenjima ostalih zajedničkih funkcionalnosti. Modul za registraciju sastoji se od tri

dijela: upravljačke logike registracije, modula za registraciju usluga i modula za registraciju korisnika. Zahtjev za registracijom korisnika ili usluge preuzima upravljačka logika registracije (1). Upravljačka logika zatim usmjerava zahtjev za registracijom na odgovarajući modul. U slučaju zahtjeva za registracijom usluge zahtjev se preusmjerava na modul za registraciju usluga (2a), dok se u slučaju zahtjeva za registracijom korisnika zahtjev preusmjerava na modul za registraciju korisnika (2b). Modul za registraciju korisnika prikuplja podatke o korisniku, kao što su korisničko ime, zaporka, ime i prezime korisnika i slično. Modul za registraciju usluga prikuplja podatke o usluzi.

Modul za registraciju usluga sastoji se od dva dijela: modula za prikupljanje osnovnih podataka o usluzi i modula za zadavne pravila praćenja usluga. Modul za prikupljanje osnovnih podataka o usluzi prikuplja podatke kao što su: naziv usluge, URL usluge, opis usluge i slično. Modul za zadavne pravila praćenja usluga omogućuje administratoru usluge definiranje dijelova korisničkih zahtjeva i odgovora poslužitelja od značaja za uslugu. Definirani dijelovi izdvajati će se iz poruka korisničkih zahtjeva i poslužiteljskih odgovora tijekom praćenja korištenja usluge na MidArc posredniku. Podatci prikupljeni za vrijeme registracije spremaju se na MidArc posredniku (3). Modul za registraciju ostvaren je kao Web usluga s grafičkim sučeljem.

Modul za registraciju usluga razvijen je za registraciju Web Services usluga. U sklopu praktičnog dijela ovog rada, modul za registraciju usluga unaprijeđen je kako bi odgovarao potrebama registracije WS-Resources usluga. Modul za prikupljanje osnovnih podataka o usluzi ostao je nepromijenjen. Modul za zadavanje pravila praćenja korištenja usluga nadograđen je kako bi odgovarao potrebama zadavanja pravila praćenja korištenja WS-Resources usluga. Budući da WS-Resources standard predstavlja proširenje Web Services standarda, nadograđeni modul može se koristiti i za zadavanje pravila praćenja korištenja Web Services usluga.



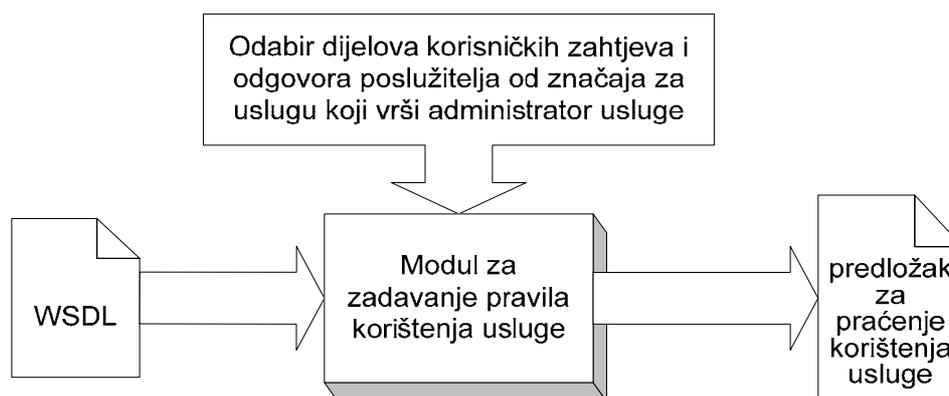
Slika 28. Organizacija registracije u MidArc posredniku.

5.3.1. Modul za zadavanje pravila praćenja korištenja usluga

Modul za zadavanje pravila praćenja korištenja usluga omogućuje administratoru svake pojedine usluge izbor dijelova korisničkog zahtjeva i odgovora poslužitelja usluge koje je potrebno izdvojiti u sklopu praćenje korištenja usluge. Izbor se vrši putem grafičkog sučelja ostvarenog kao Web usluga.

Princip rada modula za zadavanje pravila praćenja korištenja usluga prikazan je na slici 29. Ulazni parametri u modul su WSDL dokument koji opisuje uslugu, te odabir dijelova komunikacijskih poruka za praćenje koji vrši administrator usluge. Iz WSDL dokumenta vidljiva je struktura komunikacijskih poruka koje se izmjenjuju između korisnika i poslužitelja usluge. Administrator usluge odabire dijelove navedenih poruka koje treba izdvojiti u sklopu praćenja korištenja usluge. Izlazni parametar iz modula za zadavanje pravila praćenja korištenja usluga je predložak za praćenje korištenja usluge (engl. *Service Account Format*). Predložak za praćenje korištenja usluge je XML dokument koji sadrži pravila za praćenje korištenja usluge. Nakon

uspješno završene registracije usluge predložak za praćenje korištenja usluge sprema se na MidArc posredniku zajedno s ostalim podacima prikupljenim za vrijeme registracije usluge. Format predloška za praćenje korištenja usluge opisan je u poglavlju 5.3.2.

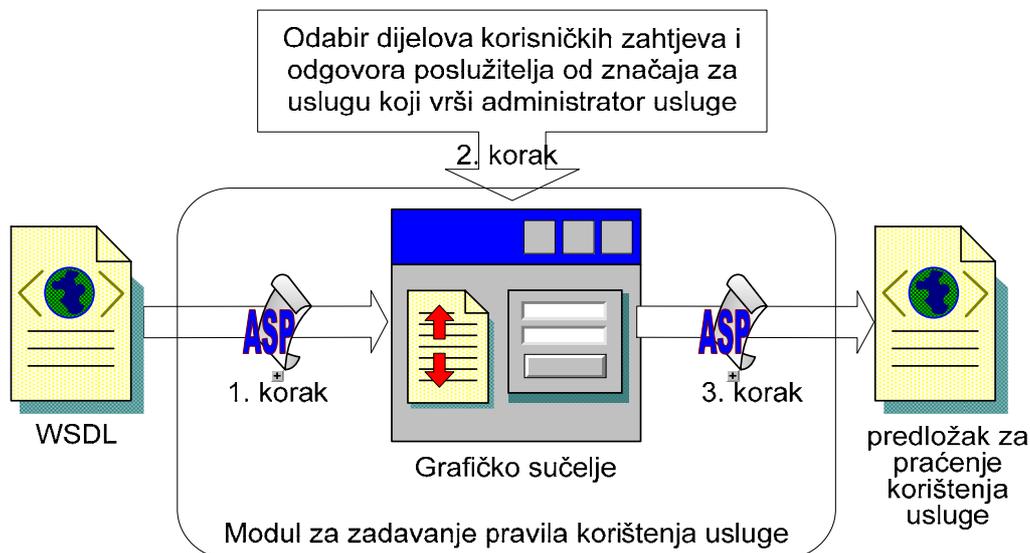


Slika 29. Princip rada modula za zadavanje pravila korištenja usluge.

Funkcionalnost modula za zadavanje pravila praćenja korištenja usluga izvodi se u tri koraka, kao što je prikazano na slici 30. U prvom koraku, na osnovu WSDL dokumenta koji opisuje uslugu izgrađuje se grafički prikaz strukture poruka koje se izmjenjuju između korisnika i poslužitelja usluge. U drugom koraku, administrator usluge putem grafičkog sučelja odabire dijelove poruka za praćenje na MidArc posredniku. Odabrani dijelovi komunikacijskih poruka izdvajat će se tijekom praćenja korištenja usluge. U trećem koraku, na osnovu navedenog odabira dijelova poruka izgrađuje se predložak za praćenje korištenja usluge.

WS-Resources standardom proširena je struktura poruka koje se izmjenjuju između korisnika i poslužitelja usluge u odnosu na Web Services standard. Nadogradnjom modula za zadavanje pravila praćenja korištenja usluga administratorima usluga omogućen je odabir dijelova poruka strukturiranih prema WS-Resources standardu. Struktura predloška za praćenje korištenja usluga razvijena za Web Services usluge [22] ne odgovara potrebama zapisivanja pravila praćenja korištenja WS-Resources usluga. Struktura predloška za praćenje korištenja usluga promjenjena je kako bi se omogućilo zapisivanje pravila praćenja WS-Resources usluga. U skladu s navedenim

promjenama, nadograđena je logika modula za zadavanje pravila praćenja korištenja usluga.



Slika 30. Arhitektura modula za zadavanje pravila korištenja usluge.

5.3.2. Struktura predloška za praćenje korištenja usluge

Predložak za praćenje korištenja usluge je XML dokument koji sadrži pravila za praćenje korištenja usluge. Pravila praćenja korištenja usluga čine podatci o dijelovima poruka korisničkih zahtjeva i odgovora poslužitelja koji su odabrani za praćenje na MidArc posredniku. Dijelovi odabrani za praćenje izdvajaju se iz poruka korisničkih zahtjeva i poslužiteljskih odgovora tijekom praćenja korištenja usluge na MidArc posredniku.

Struktura predloška za praćenje korištenja usluge prikazana je na slici 31. Za svaku WS-Resources uslugu navode se sve udaljene procedure u sklopu te usluge. Pravila praćenja zasebno su specificirana za poruku korisničkog zahtjeva, te za poruku odgovora poslužitelja usluge za svaku udaljenu proceduru. Za poruku korisničkog zahtjeva, te za poruku odgovora poslužitelja usluge zasebno se specificiraju pravila praćenja za zaglavlje, te za tijelo izmijenjenih poruka.

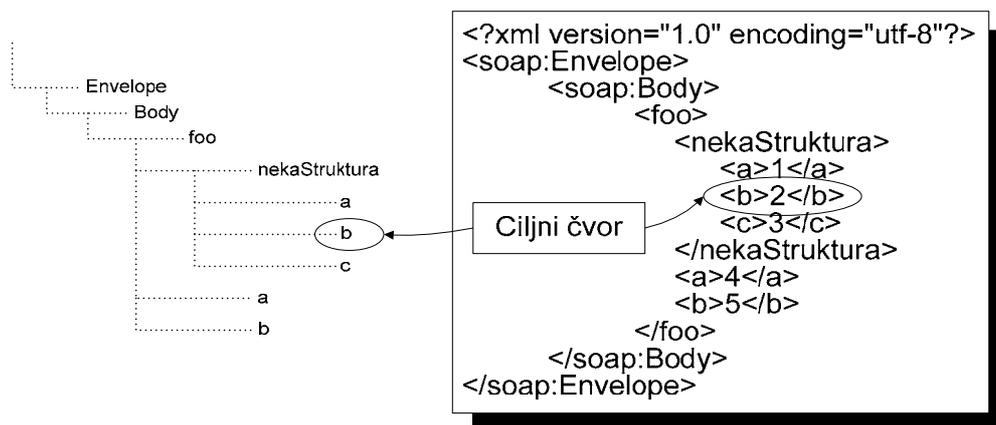


Slika 31. Struktura SAF dokumenta.

Struktura podataka u izmijenjenim porukama opisana je pomoću SOAP protokola, a prikladno ju je opisati pomoću stabla (engl. *tree*). Pojedini dio stukture podataka opisane stablom moguće je jednoznačno odrediti navođenjem staze (engl. *path*). Staza je tekstualni zapis (engl. *string*) koji se sastoji od naziva svih čvorova stabla koje je potrebno obići da bi se došlo od korijenskog do ciljnog čvora. Nazivi pojedinih čvorova obično se razdvajaju kosom crtom (/). Dio poruke koji je potrebno izdvojiti u sklopu praćenja korištenja usluge na MidArc posredniku specificira se navođenjem

staze do navedenog dijela poruke. Budući da se u predlošku za praćenje korištenja usluge pravila praćenja navode odvojeno za zaglavlje i tijelo poruke, korijenskim čvorovima stabla smatraju se čvorovi koji označavaju zaglavlje, odnosno tijelo poruke.

Na slici 32 prikazan je primjer SOAP poruke korisničkog zahtjeva, te ista poruka grafički prikazana stablom. Označeni čvor stabla sa slike jednoznačno je određen stazom koja se sastoji od naziva svih čvorova koje je potrebno obići da bi se došlo od korijenskog do označenog čvora. Budući da se čvor *Body*, koji označava tijelo poruke, smatra korijenskim čvorom, njegov se naziv ne navodi u stazi. Za poruku prikazanu na slici staza koja jednoznačno određuje označeni čvor je *foo/nekaStruktura/b*.



Slika 32. SOAP poruka. Staza do označenog dijela poruke je *foo/nekaStruktura/b*.

Opisana metoda određivanja pojedinog dijela poruke može se usporediti s metodom koja se koristi za određivanje pojedine datoteke u datotečnom sustavu. SOAP poruke kojima komuniciraju korisnik i poslužitelj usluge su XML dokumenti. Princip korištenja staze kao metode jednoznačnog određivanja pojedinih dijelova XML dokumenata standardiziran je u sklopu W3C-a pod nazivom XPath [23].

5.3.3. Grafičko sučelje za zadavanje pravila praćenja korištenja usluga

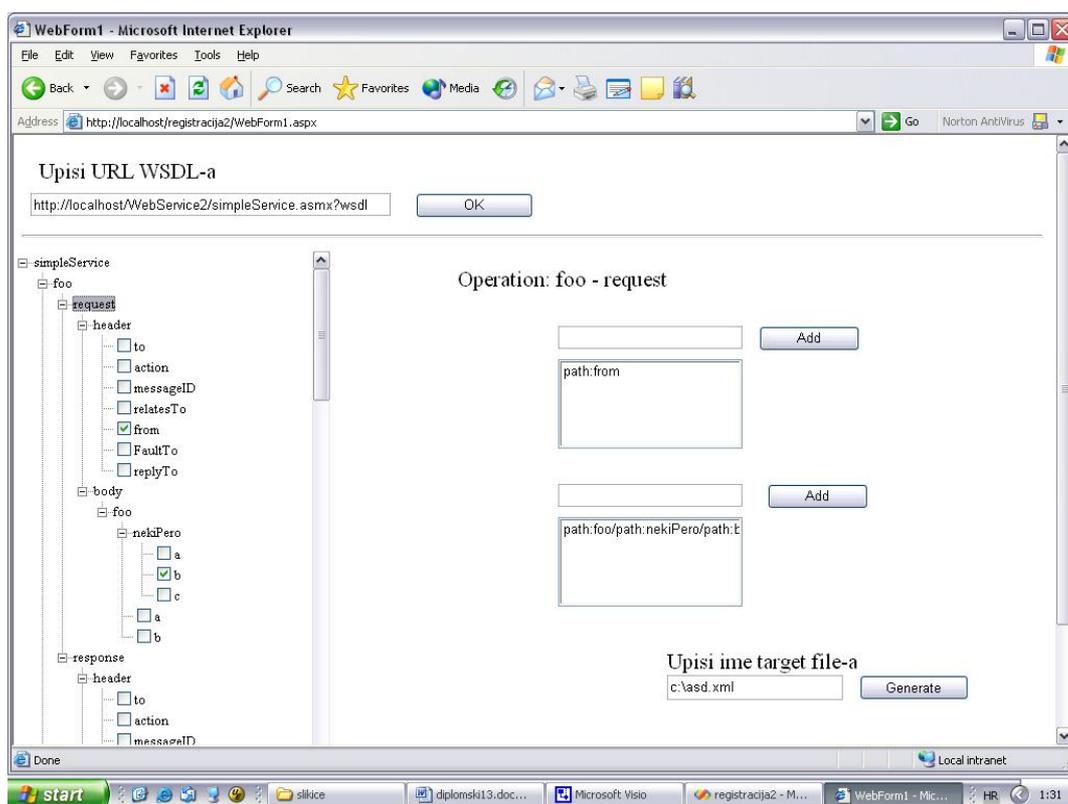
Grafičko sučelje modula za zadavanje pravila praćenja usluga omogućava administratoru usluge odabir pojedinih dijelova poruka koji će se izdvajati tijekom praćenja korištenja usluge na MidArc posredniku. Omogućene su dvije metode odabira: metoda odabira upisom staze i metoda odabira između ponuđenih dijelova poruke grafički prikazanih stablom.

Na grafičkom sučelju modula za zadavanje pravila praćenja usluga, pomoću stabla je prikazana struktura poruka korisničkog zahtjeva i odgovora poslužitelja usluge za svaku udaljenu proceduru u sklopu usluge. Administrator usluge odabirom čvorova stabla odabire dijelove poruka za praćenje na MidArc posredniku. Prednost metode odabira između ponuđenih dijelova poruke grafički prikazanih stablom je njezina jednostavnost. Međutim, nedostatak metode odabira između ponuđenih dijelova poruke predstavljaju ograničene mogućnosti specificiranja pojedinih dijelova zaglavlja poruka. Navedni nedostatak javlja se zbog činjenice da se stablo kojim su grafički prikazani dijelovi poruka gradi na osnovu WSDL dokumenta. WSDL dokumenti ne sadrže opis strukture zaglavlja, već isključivo tijela poruka. Razlog tomu je što je WSDL standard nastao kako bi se zadovoljila potreba za standardnim načinom opisivanja Web Services usluga. Budući da Web Services standard ne propisuje način korištenja zaglavlja SOAP poruka u komunikaciji korisnika i usluge, WSDL standardom opisuje se isključivo struktura tijela SOAP poruke. WS-Resources standard propisuje način korištenja zaglavlja SOAP poruka u komunikaciji korisnika i usluge. WSDL standard ne zadovoljava potrebe za opisom WS-Resources usluga. Budući da ne postoji odgovarajući standard kojim bi se opisivale WS-Resources usluge, u navedenu svrhu koristi se WSDL standard. Modul za zadavanje pravila praćenja korištenja usluga na osnovu zapisa u WSDL dokumentu izgrađuje se stablo s prikazom strukture tijela poruka. Osim strukture tijela poruka stablo prikazuje i nekoliko osnovnih dijelova zaglavlja poruka.

Za detaljnu specifikaciju dijelova zaglavlja poruka koje je potrebno pratiti na MidArc posredniku koristi se metoda odabira upisom staze. Administrator usluge odbire za praćenje dijelove poruka upisom staza do odabranih dijelova poruka. Broj staza koje

administrator usluge može upisati nije ograničen. Staze se upisuju zasebno za poruku korisničkog zahtjeva, te za poruku odgovora poslužitelja za svaku udaljenu proceduru u sklopu usluge. Osim dijelova zaglavlja poruka, metodom upisa staze mogu se odabirati i dijelovi tijela poruka.

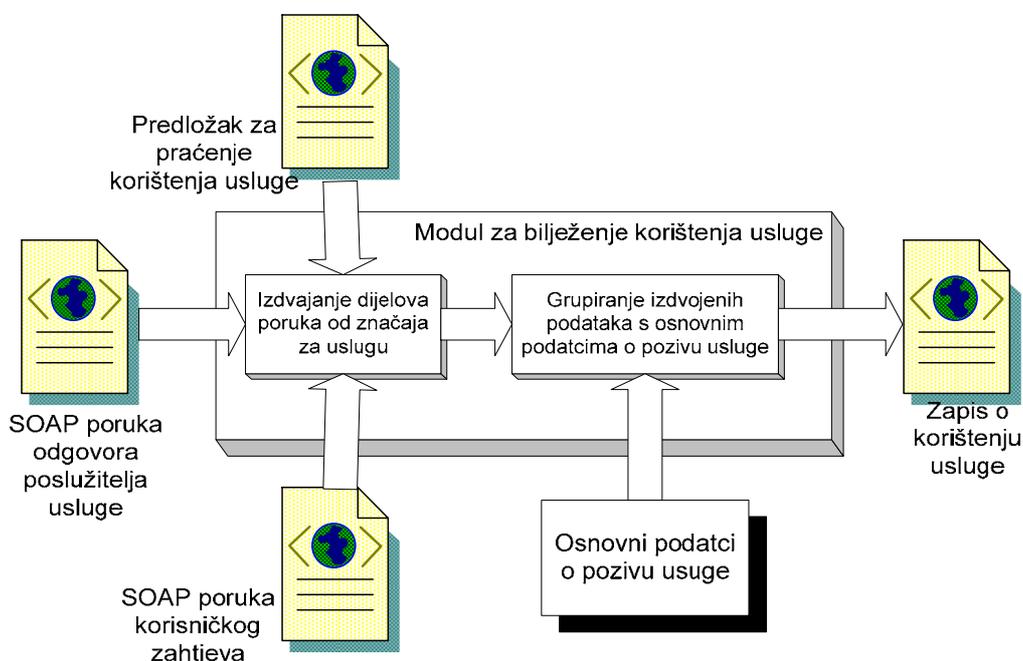
Grafičko sučelje modula za zadavanje pravila praćenja usluga razvijeno je kao Web usluga korištenjem ASP.NET tehnologije. Sastoji se od tri dijela (slika 33). Gornji dio sučelja namijenjen je upisu URL-a WSDL dokumenta koji opisuje uslugu koja se registrira. Sastoji se od okvira za upis teksta, te od tipke za potvrđivanje unosa. Lijevi dio grafičkog sučelja namijenjen je odabiru između ponuđenih dijelova poruka. Stablom je prikazana struktura poruka za komunikaciju s poslužiteljem usluge. Desni dio grafičkog sučelja namijenjen je odabiru dijelova poruku upisom staza. Odabir dijelova poruka koji će se izdvajati tijekom praćenja korištenja usluge na MidArc posredniku potvrđuje se pritiskom na tipku *Generate*.



Slika 33. Grafičko sučelje modula za zadavanje pravila praćenja usluga.

5.4. Modul za bilježenje korištenja usluga

Modul za praćenje korištenja usluga izdvaja dijelove odabrane za praćenje iz poruka korisničkog zahtjeva i odgovora poslužitelja usluge na osnovu pravila iz odgovarajućeg predloška za praćenje korištenja usluge. Izdvojeni podatci zatim se grupiraju s osnovnim podacima o pozivu usluge u zapis o korištenju usluge, te se spremaju na MidArc posredniku. Ulazni parametri u modul su SOAP poruke korisničkog zahtjeva i odgovora poslužitelja, osnovni podatci o pozivu, te predložak za praćenje korištenja usluge, a izlazni parametar je zapis o korištenju usluge (slika 34).



Slika 34. Princip rada modula za bilježenje korištenja usluga.

Unaprjeđenjem podsustava za praćenje korištenja usluga na MidArc posredniku promijenjene su strukture predloška za praćenje korištenja usluge, te zapisa o korištenju usluge. Strukture navedenih zapisa promijenjene su kako bi odgovarale potrebama praćenja korištenja WS-Resources usluga. Promjenom struktura navedenih zapisa javila se potreba za promjenom logike modula za bilježenje korištenja usluga. U sklopu praktičnog dijela ovog rada razvijen je modul za bilježenje korištenja usluga koji u svojem radu koristi zapise promijenjene strukture. Izdvajanje dijelova poruka određenih za praćenje izvodi se korištenjem predloška za praćenje

korištenja usluga opisanog u poglavlju 5.3.2. Izlazni parametar modula za bilježenje korištenja usluga je zapis o korištenju WS-Resources usluge. Razvijeni modul za bilježenje korištenja WS-Resources usluga zamjenjuje modul za praćenje korištenja Web Services usluga. Budući da WS-Resources standard predstavlja proširenje Web Services standarda, razvijeni modul može se koristiti i za bilježenje korištenja Web Services usluga.



Slika 35. Struktura zapisa o korištenju usluge.

Zapis o korištenju WS-Resources usluge sadrži podatke o jednom pozivu usluge. Struktura zapisa prikazana je na slici 35. Sastoji se od osnovnih podataka o pozivu usluge, te od zapisa o korisničkom zahtjevu i zapisa o odgovoru poslužitelja usluge. Osnovne podatke o pozivu usluge čine: identifikator korisnika, identifikator usluge, identifikator pozvane procedure, identifikator autoriziranosti poziva usluge, te vrijeme

upućivanja zahtjeva i odgovora. Identifikator korisnika i identifikator usluge su jedinstvene oznake dodijeljene prilikom registracije korisnika, odnosno usluge. Korištenje WS-Resources usluge predstavlja poziv udaljene procedure. Identifikator pozvane procedure predstavlja naziv udaljene procedure pozvane prilikom korištenja usluge. Identifikator autoriziranosti sadrži podatak tome je li korisnik imao odgovarajuća prava za poziv usluge.

Zapis o korisničkom zahtjevu sastoji se od podatka o vremenu upućivanja zahtjeva, te od elemenata izdvojenih iz poruke korisničkog zahtjeva. Vrijeme upućivanja zahtjeva predstavlja trenutak u kojem je korisnički zahtjev stigao na MidArc posrednik. Elementi izdvojeni iz poruke korisničkog zahtjeva dijele se na elemente izdvojene iz zaglavlja, te one izdvojene iz tijela poruke. Zapis o odgovoru poslužitelja usluge sastoji se od podatka o vremenu upućivanja odgovora, te od elemenata izdvojenih iz poruke odgovora poslužitelja. Vrijeme upućivanja odgovora predstavlja trenutak u kojem je odgovor poslužitelja stigao na MidArc posrednik. Elementi izdvojeni iz poruke odgovora poslužitelja dijele se na elemente izdvojene iz zaglavlja, te one izdvojene iz tijela poruke.

Modul za praćenje korištenja usluga ostvaren je u .NET tehnologiji. Izvorni kod napisan je u jeziku C#. Modul je ostvaren kao razred *UsageRecord*. Dijelovi zapisa o korištenju usluge, predstavljaju članske (engl. *member*) varijable razreda *UsageRecord*. Postavljanje odgovarajućih vrijednosti u navedene varijable provodi se pozivom članske funkcije *Account*.

6. Zaključak

Posrednički sustavi omogućavaju brži i jednostavniji razvoj Internet usluga. Ugradnjom zajedničkih funkcionalnosti u posrednički sustav rasterećuje se programere pri razvoju usluga. Umjesto razvoja zajedničkih funkcionalnosti za svaku pojedinu uslugu iznova, one su razvijene jednom, te se nalaze na posredniku. Usluge se u svom radu za pružanje zajedničkih funkcionalnosti oslanjaju na posrednika. Suradnjom Ericsson Nikola Tesle i Zavoda za Elektroniku, Mikroelektroniku, Računalne i Inteligentne Sustave Fakulteta Elektrotehnike i Računarstva u Zagrebu razvijen je posrednički sustav MidArc. MidArc posrednik pruža zajedničke funkcionalnosti registracije, autentikacije, autorizacije i praćenja korištenja usluga, te osigurava sigurnost i povjerljivost komunikacije. Zajednička funkcionalnost praćenja korištenja usluga prikuplja podatke o korištenju usluga putem MidArc posrednika. Podatci prikupljeni praćenjem korištenja usluga ustupaju se uslugama na analizu. Rezultati analize prikupljenih podataka najčešće se koriste za naplatu, izradu korisničkih profila i slično.

Funkcionalnost praćenja korištenja usluga putem MidArc posrednika razvijena je za praćenje korištenja Web Services usluga. U međuvremenu, razvijen je novi standard pružanja usluga pod nazivom WS-Resources. Razvojem WS-Resources tehnologije javlja se potreba za omogućavanjem praćenja korištenja pojedinog sredstva usluge putem MidArc posrednika. U sklopu ovog rada izvršena je analiza razlika između navedenih tehnologija. Proučene su metode kojima se danas na Internetu prati korištenje usluga, te posebno metoda kojom se u sklopu MidArc posrednika prati korištenje Web Services usluga. U skladu s uočenim potrebama, unaprijeđen je podsustav MidArc posrednika za praćenje korištenja usluga. Nadograđen modul za zadavanje pravila za praćenje korištenja usluga pri registraciji usluge, te je razvijen modul za bilježenje korištenja WS-Resources usluga. WS-Resources standard predstavlja proširenje Web Services standarda, te se razvijeni moduli mogu koristiti i za praćenje korištenja Web Services usluga.

Podsustav MidArc posrednika za praćenje korištenja usluga omogućava detaljnu specifikaciju pravila praćenja korištenja WS-Resources usluga. Tijekom budućeg rada moguće je funkcionalnost praćenja korištenja usluga

proširiti kako bi se omogućila detaljna specifikacija pravila praćenja pristupa Web stranicama. Daljnji razvoj praćenja korištenja usluga može se usmjeriti prema nadgledanju rada i komunikaciji usluga unutar sustava. Unutarnjim nadgledanjem rada sustava može se dobiti uvid u svojstva sustava, opterećenja, probleme u komunikaciji i funkcionalnosti dijelova sustava. U slučaju razvoja novih tehnologija pružanja usluga, potrebno je prilagođavanje funkcionalnost praćenja korištenja usluga radu s novim tehnologijama.

Literatura

- [1] A. S. Tanenbaum: „Computer Networks, Fourth Edition“, Prentice Hall, 2003.
- [2] Simple Mail Transfer Protocol (SMTP), RFC 2821, Network Working Group, <http://www.faqs.org/rfcs/rfc2821.html>, travanj 2001.
- [3] Post Office Protocol - Version 3 (POP3), RFC 1939, Network Working Group, <http://www.faqs.org/rfcs/rfc1939.html>, svibanj 1996.
- [4] File Transfer Protocol (FTP), RFC 959, Network Working Group, <http://www.faqs.org/rfcs/rfc959.html>, listopad 1985.
- [5] J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: „Hypertext Transfer Protocol – HTTP/1.1“, The Internet Society, <http://www.w3.org/Protocols/rfc2616/rfc2616.html>, 1999.
- [6] Extensible Markup Language (XML) 1.0 (Second Edition), World Wide Web Consortium, <http://www.w3.org/TR/REC-xml>, 2000.
- [7] SOAP Version 1.2, World Wide Web Consortium, <http://www.w3.org/TR/2003/REC-soap12-part0-20030624>, 2003.
- [8] Web Services Description Language (WSDL) 1.1, World Wide Web Consortium, <http://www.w3.org/TR/wsdl.html>, 2001.
- [9] UDDI Technical White Paper, Ariba, Inc., International Business Machines Corporation, Microsoft Corporation, http://www.uddi.org/pubs/lru_UDDI_Technical_White_Paper.pdf, 2000.
- [10] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, P. Vanderbilt: „Open Grid Services Infrastructure“, <http://www.ggf.org/ogsi-wg>, 2003.

-
- [11] I. Sučić: „Grid“, Seminarski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2002.
- [12] I. Foster, C. Kesselman, S. Tuecke: „The Anatomy of the Grid: Enabling Scalable Virtual Organizations“, International J. Supercomputer Applications, 2001.
- [13] I. Foster, C. Kesselman, S. Tuecke, J. M. Nick: „The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration“, Globus Project, 2002.
- [14] Karl Czajkowski, Donald F Ferguson, Ian Foster, Jeffrey Frey, Steve Graham, Igor Sedukhin, David Snelling, Steve Tuecke, William Vambenepe: „The WS-Resource Framework, Version 1.0“, ožujak 2004.,
- [15] Ian Foster, Jeffrey Frey, Steve Graham, Steve Tuecke, Karl Czajkowski, Don Ferguson, Frank Leymann, Martin Nally, Igor Sedukhin, David Snelling, Tony Storey, William Vambenepe, Sanjiva Weerawarana: „Modeling Stateful Resources with Web Services, Version 1.1“, ožujak 2004.
- [16] Adam Bosworth, Don Box, Erik Christensen, Francisco Curbera, Donald Ferguson, Jeffrey Frey, Chris Kaler, David Langworthy, Frank Leymann, Brad Lovering, Steve Lucco, Steve Millet, Nirmal Mukhi, Mark Nottingham, David Orchard, John Shewchuk, Tony Storey, Sanjiva Weerawarana: „Web Services Addressing (WS-Addressing)“, ožujak 2004.
- [17] R. Kimball, R. Merz: „The Data Webhouse Toolkit“, Wiley Computer Publishing, 2000.
- [18] I. Benc: „Gospodarenje podacima u posredniku javnog informacijskog sustava“, Magistarski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2003.
- [19] D. Krišto: „Registracija korisnika i mrežnih usluga Middleware sustava“, Diplomski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2002.

- [20] M. Popović: „Priručna memorija posredničkog sustava“, Diplomski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2003.
- [21] M. Štefanec: „Zajedničke usluge posredničkog sustava u globalnoj mreži Internet“, Magistarski rad, Fakultet elektrotehnike i računarstva, Zagreb, 2003.
- [22] I. Benc, M. Štefanec, S. Srblijić: „Usage Tracking by Public Information System Mediator“, MELECON, Zagreb, 2003.
- [23] XML Path Language (XPath) 1.0, World Wide Web Consortium, <http://www.w3.org/TR/xpath>, studeni 1999.