SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1663

ISPITIVANJE RADNIH ZNAČAJKI USLUGA U RAZVOJNOJ OKOLINI PIE

Vedran Najrajter

Zagreb, rujan 2007.

Zahvaljujem prof. dr. sc. Siniši Srbljiću na prilici za rad u poticajnoj okolini.

Zahvaljujem mr. sc. Danielu Skrobi na trudu, strpljenju i pomoći pri izradi ovog rada.

Veliko hvala roditeljima, sestri i prijateljima na podršci tijekom studija.

"Dobro je da čovjek sam iskusi sve što treba da zna."
H. Hesse

S A D R Ž A J

1 Uvod

Razvoj programskih tehnologija za izgradnju sustava zasnovanih na uslugama omogućio je izgradnju računalnih sustava velikih razmjera (engl. *large scale*). Budući da se računalni sustavi velikih razmjera izvode u raznorodnim (engl. *heterogeneous*) računalnim okolinama i zemljopisno su rasprostranjeni, postupak upravljanja radom sustava vrlo je složen. S namjerom olakšavanja postupka upravljanja grade se specijalizirani sustavi koji omogućuju ispitivanje radnih značajki raspodijeljenih sustava. Sustavi za ispitivanje radnih značajki služe kao pripomoć upraviteljima sustava i omogućavaju prikupljanje informacija o računalnoj i mrežnoj infrastrukturi na temelju kojih upravitelji sustava otkrivaju nedostatke i nepravilnosti u radu raspodijeljenog sustava.

U diplomskom radu opisan je model sustava za ispitivanje radnih značajki sustava zasnovanih na uslugama i njegovo programsko ostvarenje. Opisani model sustava za ispitivanje radnih značajki zasnovan je na raspodijeljenoj arhitekturi koja se sastoji od upravljačkog podsustava, podsustava za poziv usluga i podsustava za mjerenje radnih značajki. Ostvareni sustav za ispitivanje radnih značajki moguće je koristiti za prikupljanje informacija o zauzeću raspodijeljenih računalnih sredstava na temelju kojih se dobiva uvid u radne značajke ispitivanog sustava zasnovanog na uslugama. Analizom prikupljenih radnih značajki moguće je otkriti nepravilnosti i nedostatke u radu sustava te njihovim uklanjanjem poboljšati rad ispitivanog sustava.

U drugom poglavlju opisana su načela računarstva zasnovanog na uslugama i ostvaren je pregled programskih tehnologija i standarda koji se koriste pri izradi sustava zasnovanih na uslugama. Treće poglavlje opisuje arhitekturu sustava za ispitivanje radnih značajki i postojeće sustave za ispitivanje radnih značajki dostupne na tržištu. Četvrto poglavlje opisuje razvojnu okolinu PIE koja omogućava izgradnju i izvođenje primjena u globalnoj mreži Internet. Peto poglavlje opisuje Sustav za nadgledanje rada korisnika i usluga čije su radne značajke ispitivane primjenom ostvarenog sustava. U šestom poglavlju opisan je model Sustava za ispitivanje radnih značajki i njegovo programsko ostvarenje. Sedmo poglavlje prikazuje rezultate ispitivanja Sustava za nadgledanje rada korisnika i usluga primjenom Sustava za ispitivanje radnih značajki. U osmom poglavlju naveden je zaključak diplomskog rada.

2 Računarstvo zasnovano na uslugama

Razvoj računalnih mreža omogućio je povezivanje programskih sustava u složene raspodijeljene računalne sustave. Kako bi se olakšalo oblikovanje i razvoj raspodijeljenih sustava, definirane su različite programske paradigme. U samim počecima razvoja, programski sustavi su se gradili kao monolitni programi. Porast složenosti programske podrške dovodi do razlaganja monolitnih programa na manje dijelove i organizacije programskog koda u module. Postupak oblikovanja programa razlaganjem programskog koda u manje dijelove naziva se proceduralna programska paradigma. Proceduralno programiranje imalo je veliki nedostatak. Naime, održavanje programske podrške bio je zahtjevan posao budući da je promjena u jednom od modula zahtijevala provođenje promjena u svim programskim modulima koji su koristili programski kod tog modula. Proceduralnu programsku paradigmu zamjenjuje objektno-orijentirana paradigma čijom se upotrebom izgrađuju programi kao skupine objekata čije su značajke definirane različitim razredima. Razred predstavlja model koji nastaje apstrakcijom značajki objekata iz stvarnog svijeta. Višestruka iskoristivost zasniva se na stvaranju primjeraka razreda, objekata. Promjene koje se uvode u razredima ne utječu na korisnike ukoliko se ne mijenja sučelje razreda. Prednosti objektno orijentirane paradigme vidljive su tijekom razvoja i za vrijeme održavanja programske podrške. Radi ubrzavanja razvoja programske podrške uvodi se programska paradigma zasnovana na komponentama. Komponenta je dio programskog sustava koja obavlja skup dobro definiranih funkcionalnosti koje su dostupne putem programskog sučelja komponente. Upotrebom gotovih komponenti drugih proizvođača ubrzan je postupak izgradnje programskih sustava. Programska paradigma zasnovana na komponentama pokazuje svoje nedostatke prilikom oblikovanja programskih sustava objedinjavanjem postojećih i novonastalih programskih sustava koji su izgrađeni upotrebom raznorodnih programskih tehnologija. Rješenje problema objedinjavanja raznorodnih programskih tehnologija ponudila je programska paradigma zasnovana na uslugama.

Primjenom programske paradigme zasnovane na uslugama grade se programski sustavi kao skupine povezanih programskih usluga. Programsku paradigmu zasnovanu na uslugama odlikuje neovisnost programskih usluga o programskim tehnologijama kojima su ostvarene njihove funkcionalnosti i tehnološkim uvjetima koji vladaju u okolini u kojoj se izvode. Spomenuta neovisnost

zasniva se na primjeni standardnih tehnologija za izgradnju i objedinjavanje sustava zasnovanih na uslugama. Računarstvo zasnovano na uslugama izučava metodologiju oblikovanja, izgradnje, povezivanja i korištenja raspodijeljenih računalnih sustava čija je osnovna jedinica izgradnje programska usluga. Računalni sustavi zasnovani na uslugama grade se na načelima arhitekture zasnovane na uslugama.

2.1 Programska usluga

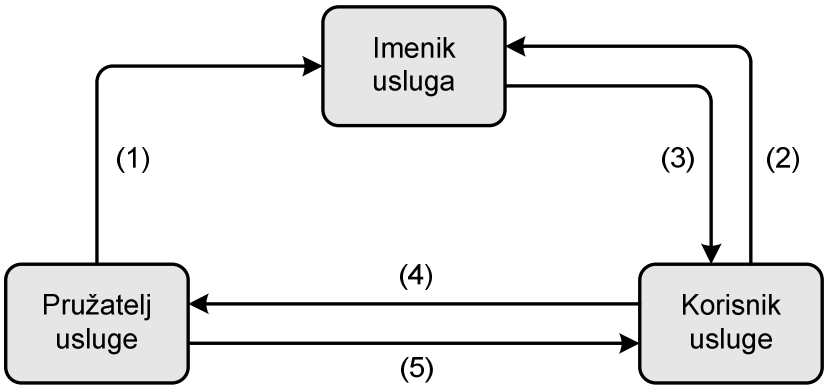
Programska usluga je samostojeća i neovisna programska jedinica s dobro definiranom funkcionalnosti. Razvijatelj sustava gradi programsku uslugu i postavlja ju na pružatelja usluga koji osigurava dostupnost, izvođenje i održavanje programskih usluga. Programsku uslugu karakterizira odvajanje poslovne logike i pristupnog sučelja putem kojeg programska usluga izlaže svoju funkcionalnost. Poslovna logika programske usluge koja ostvaruje funkcionalnost može biti izgrađena upotrebom različitih programskih tehnologija koje mogu biti međusobno neuskladive, smještena na jednom ili više čvorova i izvoditi se u okolini s proizvoljnim operacijskim sustavom. Izgradnja pristupnih sučelja programskih usluga izvršava se upotrebom otvorenih i standardiziranih programskih tehnologija. Opis pristupnog sučelja sadrži informacije o funkcionalnosti programske usluge, opis strukture poruka koje se razmjenjuju s programskom uslugom tijekom poziva usluge i tehničke informacije potrebne za povezivanje (engl. *bind*) na programsku uslugu. Dokument s opisom pristupnog sučelja programske usluge gradi vlasnik programske usluge i zatim ga objavljuje u imeniku usluga. Objavljivanjem opisa pristupnog sučelja omogućava se otkrivanje i korištenje programske usluge. Programska usluga je u potpunosti određena pristupnom točkom računalne mreže na kojoj je dostupna, pristupnim sučeljem i funkcionalnošću koju izlaže putem pristupnog sučelja.

Osnovna jedinica izgradnje sustava zasnovanih na uslugama je programska usluga. Povezivanjem programskih usluga grade se slabo povezani (engl. *loosely coupled*) računalni sustavi. Pojam slabo povezivanje podrazumijeva uklanjanje zavisnosti programskih usluga koje proizlaze iz načina na koji programske usluge međusobno komuniciraju. Slabo povezivanje programskih usluga zasniva se na mogućnosti otkrivanja i povezivanja programskih usluga tijekom izvođenja programskog sustava zasnovanog na uslugama. Povezivanje tijekom izvođenja

(engl. *dynamic binding*) moguće je jedino uz korištenje otvorenih i standardnih tehnologija za izgradnju i pristupanje sučeljima programskih usluga. Komunikacija između programskih usluga odvija se razmjenom poruka. Poruke su strukture koje prenose parametre poziva i rezultate izvođenja operacije programske usluge. Budući da razvijatelji programskih usluga koriste različite programske tehnologije za izgradnju programskih usluga komunikacija između programskih usluga moguća jedino uz standardizaciju. Stoga standardizirane i otvorene tehnologije definiraju oblik zapisa poruka i postupak razmjene poruka. Dio opisa pristupnih sučelja programskih usluga sadrži informacije o ispravnoj strukturi ulaznih i izlaznih poruka. Budući da je moguća izgradnja poruka potrebnih za komunikaciju s programskom uslugom na temelju opisa pristupnog sučelja programske usluge omogućena je izgradnja programskih usluga neovisnih o sučeljima ostalih programskih usluga s kojima je potrebno ostvariti suradnju.

2.2 Arhitektura zasnovana na uslugama

Arhitektura sustava zasnovanih na uslugama sastoji se od skupa slabo povezanih usluga. Programske usluge imaju jasno definirana pristupna sučelja koja se opisuju upotrebom otvorenih i standardiziranih programskih tehnologija. Pristupna sučelja programskih usluga sastoje se od skupa operacija koje programska usluga izlaže potencijalnim korisnicima usluge. Razvijatelj programske usluge stvara opis pristupnog sučelja programske usluge koji objavljuje i tako omogućava njezino otkrivanje. Korisnici usluga otkrivaju željenu programsku uslugu, dohvaćaju informacije o njezinom pristupnom sučelju i pozivaju je. Osnovna arhitektura sustava zasnovanih na uslugama prikazana je na slici 2.1.



Slika 2.1: Arhitektura sustava zasnovanih na uslugama

Osnovna arhitektura sustava zasnovanih na uslugama sastoji se od tri komponente: pružatelj usluge, imenik usluga i korisnik usluge. Pružatelj usluge gradi i postavlja programsku uslugu. Izgrađenu programsku uslugu nudi drugim korisnicima na korištenje objavljivanjem (1) opisa pristupnog sučelja izgrađene programske usluge u imeniku usluga. Imenik usluga predstavlja registar raspoloživih usluga koje korisnici usluga pretražuju s ciljem pronalaska programske usluge s željenom funkcionalnosti. Korisnik usluge pretraživanjem (2) imenika usluga pronalazi (3) opis pristupnog sučelja željene programske usluge. Na temelju informacija iz opisa pristupnog sučelja programske usluge korisnik usluge šalje (4) poruku zahtjeva programskoj usluzi za izvođenje operacije navedene u opisu pristupnog sučelja. Programska usluga izvodi željenu operaciju i vraća (5) rezultate izvođenja.

Povezivanje (engl. *composition*) usluga je glavni postupak u izgradnji računalnih sustava zasnovanih na uslugama. Povezivanjem postojećih programskih usluga ostvaruju se nove i složenije programske usluge. Novoizgrađene usluge moguće je koristiti u daljnjoj izgradnji složenijih usluga ili ih je moguće izložiti na korištenje potencijalnim korisnicima. Postoje dva pristupa u povezivanju programskih usluga, orkestracija (engl. *orchestration*) i koreografija (engl. *choreography*). Orkestracija usluga zasnovana je na primjeni poslovnih procesa koji se sastoje od međudjelovanja programskih usluga. Poslovna logika poslovnog procesa definira obrasce za razmjenu poruka i transformaciju poruka između programskih usluga koje su sastavni dio složene usluge. Naziv orkestracija posljedica je središnje i upravljačke uloge poslovnog procesa. Koreografija usluga temelji se na opisu uloge svake od programskih usluga koje čine sastavni dio složene programske usluge. Opis koreografije usluga ostvaruje se proširenjem opisa pristupnog sučelja svake sastavne programske usluge. Proširivanjem opisa sučelja usluge dodaje se skup naredbi za ostvarivanje koreografije. Korištenjem ovog pristupa programske usluge, koje su dio složene usluge, su ravnopravne i ne postoji središnja upravljačka komponenta.

2.3 Tehnologija Web Services

Web Services tehnologija (WS*-) definira standarde i mehanizme koji predstavljaju okvir za izgradnju raspodijeljenih računalnih sustava zasnovanih na uslugama. Web Services standardi zasnivaju se na WS-* skupu otvorenih

tehnologija. Osnovu svih tehnologija u skupu WS-* čine jezici XML i XML Schema. Temeljne tehnologije koje se koriste za razvoj raspodijeljenih računalnih sustava skupa WS-* su SOAP, WSDL i UDDI. SOAP tehnologija definira komunikacijski protokol za razmjenu podataka između primjenskih programa. WSDL tehnologija definira jezik za opisivanje pristupnih sučelja programskih usluga. UDDI tehnologija definira protokole i mehanizme za objavljivanje i pretraživanje imenika usluga.

2.3.1 XML i XML Schema

Jezik XML koristi se za opisivanje sadržaja i strukture podataka [1]. Budući da se opis podataka u potpunosti zasniva na tekstualnom zapisu XML jezik je neovisan o korištenoj programskoj tehnologiji i pogodan za prikazivanje podatkovnih struktura u različitim programskim okolinama. Svaki XML dokument ima korijenski element unutar kojeg se ugnježđuje konačan broj drugih elemenata. Svaki element može imati konačan broj atributa. Atributi opisuju elemente koji se nalaze ugniježđeni unutar elementa kojemu su ti atributi pridruženi. Svaki atribut i element jedinstveno je određen svojim imenom i prostorom imena (engl. *namespace*) kojem pripada. Zbog široke upotrebe XML dokumenata i velike vjerojatnosti pojave jednakih imena za označavanje dvaju strukturno različitih elemenata uvedeni su prostori imena. S razlogom izbjegavanja sukoba imena moguće je svakom XML dokumentu pridružiti jedinstveni prostor imena.

```
<knjižnica xmlns="svemojeknjige">
  <knjiga naziv="Kenjaža: Teorijski pristup">
    <autor>Harry G. Frankfurt</autor>
    <izdavač>Algoritam</izdavač>
    <godina>2006</godina>
  </knjiga>
  <knjiga naziv="Isusov sin">
    <autor>Denis Johnson</autor>
    <izdavač>Profil</izdavač>
    <godina>2005</godina>
  </knjiga>
</knjižnica>
```

Slika 2.2: Primjer XML dokumenta

Slika 2.2 prikazuje primjer XML dokumenta. Korijenski element je *<knjižnica>* koji ima definiran atribut *xmlns*. Vrijednost koja se pridjeljuje atributu smješta se unutar dvostrukih navodnika. Atribut *xmlns* definira naziv podrazumijevanog (engl.

default) prostora imena XML dokumenta. Unutar korijenskog elementa ugniježđena su dva elementa *<knjiga>*. Element *<knjiga>* ima definiran atribut *naziv* koji određuje ime knjige. Unutar elementa *<knjiga>* ugnježđuju se elementi *<autor>*, *<izdavač>* i *<godina>*.

Jezik XML Schema koristi se za definiranje strukture i sadržaja XML dokumenata [2]. Za definiranje sadržaja XML dokumenata na raspolaganju se nalaze standardni podatkovni tipovi poput cjelobrojnog tipa, broj s posmačnim zarezom i niz tekstualnih znakova. Korištenjem XML Schema jezika gradi se XML Schema dokument na temelju kojeg je moguće automatizirano izgraditi jezični procesor koji prihvaća opisanu klasu XML dokumenata.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="knjižnica">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="knjiga" maxOccurs="unbound">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="autor" type="xs:string">
              <xs:element name="izdavač" type="xs:string">
              <xs:element name="godina" type="xs:positiveInteger">
            </xs:sequence>
            <xs:attribute name="naziv" type="xs:string" use="required">
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

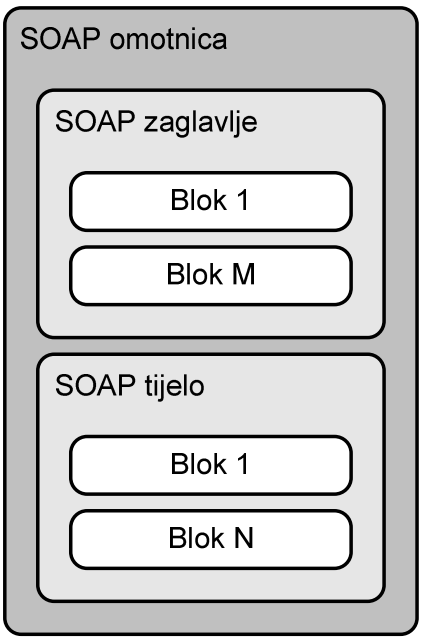
Slika 2.3: Primjer XML Schema dokumenta

Slika 2.3 prikazuje primjer XML Schema dokumenta koji definira strukturu i sadržaj XML dokumenta prikazanog na slici 2.2. Korijenski element je *<xs:schema>*. Upotrebom elementa *<xs:element>* definiraju se elementi XML dokumenta. Elementi mogu biti jednostavni ili složeni. Ako su elementi jednostavni onda imaju definirane atribute *name* i *type*, a ako su složeni onda im je prvi ugniježđeni element *<xs:complexType>*. Atribut *name* označava ime XML elementa, a *type* tip podatka koji se može pridružiti jednostavnom elementu. Element *<xs:sequence>* određuje strukturu složenog elementa kao slijed u njemu ugniježđenih elemenata. Atributi XML

elemenata definiraju se upotrebom elementa *<xs:attribute>* koji ima definirane atribute *name*, *type* i *use*. Atributi *name* i *type* imaju isto značenje kao kod elementa *<xs:element>*, dok vrijednost atributa *use* služi za definiranje obaveznog pojavljivanja pridruženog mu atributa.

2.3.2 SOAP

SOAP tehnologija definira otvoren i proširiv komunikacijski protokol za razmjenu podataka između primjenskih sustava [3]. SOAP definira pravila o ispravnom strukturiranju SOAP poruka i aktivnosti upravljanja SOAP porukama na putu od izvorišta do odredišta. SOAP poruka predstavlja osnovnu jedinicu komunikacije temeljenu na SOAP komunikacijskom protokolu. Slika 2.4 prikazuje strukturu SOAP poruke.



Slika 2.4: Struktura SOAP poruke

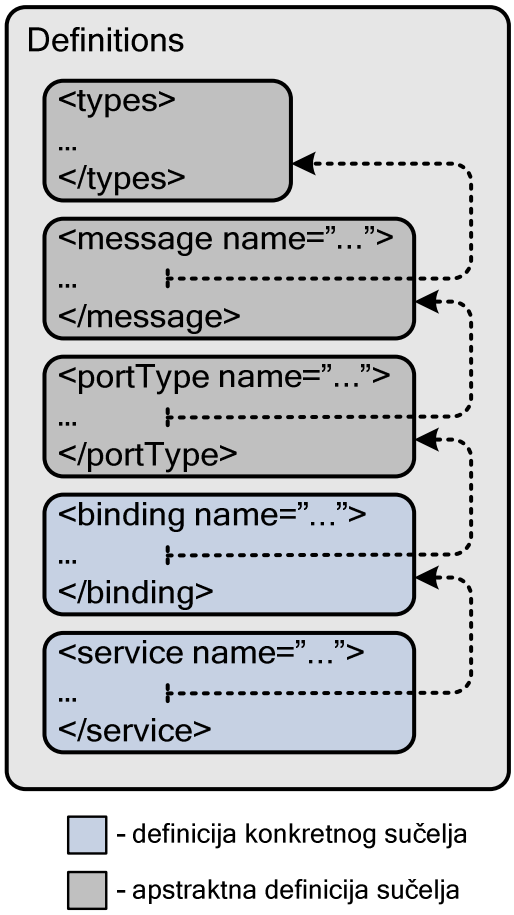
SOAP poruka sastoji se od SOAP omotnice (engl. *envelope*). SOAP omotnica sadrži zaglavlje (engl. *header*) i tijelo (engl. *body*) SOAP poruke. Zaglavlje SOAP poruke nosi upravljačke informacije poput podataka o sigurnosti, pouzdanosti i adresiranju. Dodavanjem blokova u SOAP zaglavlje moguće je definirati proizvoljne komunikacijske protokole zasnovane na SOAP protokolu i tako proširivati osnovne funkcionalnosti SOAP protokola. Tijelo SOAP poruke nosi podatke koje primjenski programi razmjenjuju. SOAP protokol omogućuje korištenje dva obrasca komunikacije zahtjev-odgovor (engl. *request-response*) i jednosmjernu komunikaciju

(engl. *one-way messaging*). Za potrebe pakiranja i prijenosa SOAP poruka moguće je koristiti proizvoljne transportne komunikacijske protokole kao što su HTTP (engl. *Hypertext Transfer Protocol*), SMTP (engl. *Simple Mail Transfer Protocol*) i TCP (engl. *Transmission Control Protocol*).

2.3.3 WSDL

WSDL (engl. *Web Services Description Language*) jezik koristi se za opisivanje pristupnih sučelja programskih usluga [4]. Opisom pristupnog sučelja programske usluge primjenom WSDL jezika vlasnik programske usluge pruža potencijalnim korisnicima informacije kako ispravno strukturirati ulazne SOAP poruke i kakva se struktura izlaznih SOAP poruka može očekivati kao rezultat izvođenja operacija programske usluge. WSDL dokument ne opisuje semantiku programske usluge i njezinih pojedinih operacija, dakle ne pruža nikakve informacije o značenju rezultata razmjene SOAP poruka s programskom uslugom.

Svaki WSDL dokument sastoji se od apstraktne definicije sučelja i definicije konkretnog sučelja. Apstraktna definicija sučelja neovisna je o programskom ostvarenju pristupnog sučelja i stoga je višestruko iskoristiva. Apstraktna definicija sučelja opisuje operativno ponašanje programske usluge nabrajanjem raspoloživih operacija pristupnog sučelja, opisom strukture ulaznih i izlaznih poruka koje se razmjenjuju pri pozivu određene operacije te opisom tipova podataka koji su dio ulaznih i izlaznih poruka. Definicija konkretnog sučelja usluge sadrži informacije o tome kako i gdje pristupiti konkretnoj implementaciji programske usluge. Slika 2.5 prikazuje strukturu WSDL dokumenta. Apstraktna definicija sučelja sadrži element *types*, skup elemenata *message* i skup elemenata *portType*. Element *types* sadrži XML Schema opise ili kazaljke na vanjske XML Schema opise podatkovnih struktura koje se koriste u WSDL dokumentu. Za opis podatkovnih struktura mogu se koristiti različiti jezici, ali se najčešće koristi XML Schema jezik. Element *message* opisuje strukturu ulaznih ili izlaznih poruka programske usluge. Za opis strukture koriste se tipovi podataka definirani u elementu *types*. U jednom WSDL dokumentu može se nalaziti konačan broj *message* elemenata. Element *portType* sadrži listu operacija pristupnog sučelja i kazaljke na elemente *message* koje operacija koristi. Element *binding* sadrži kazaljke na elemente *portType* i informacije o tome koje je transportne komunikacijske protokole potrebno koristiti i postavke kako pakirati SOAP poruke pri



Slika 2.5: Struktura WSDL dokumenta

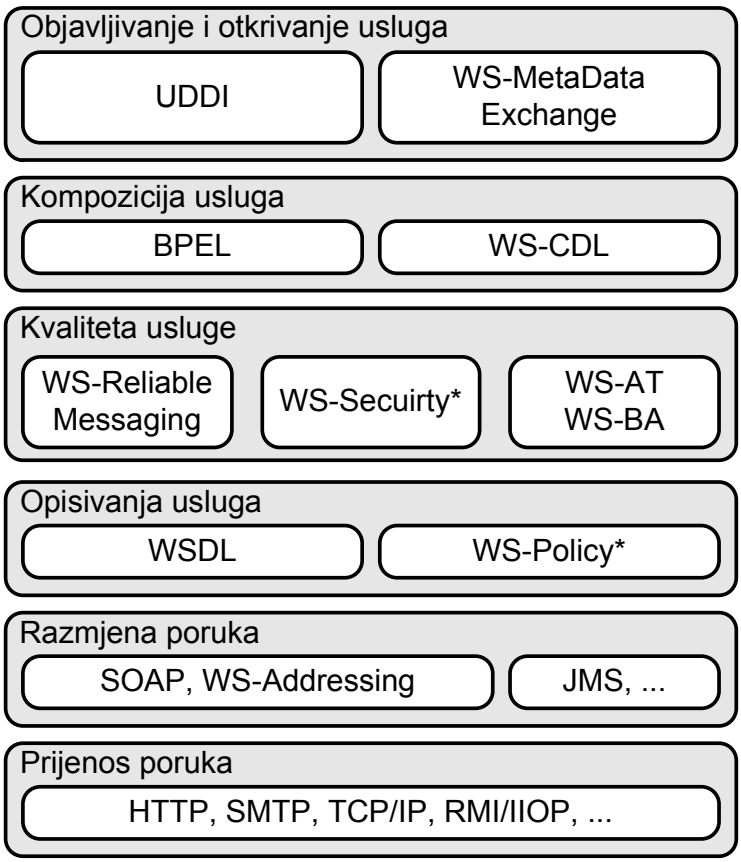
izabranom komunikacijskom protokolu. Element *service* sadrži ime programske usluge i mrežnu adresu na kojoj je programska usluga dostupna.

2.3.4 UDDI

UDDI (engl. *Universal Description, Discovery and Integration*) tehnologija definira arhitekturu imenika programskih usluga i mehanizme za objavljivanje, pretraživanje i otkrivanje programskih usluga dostupnih na globalnoj računalnoj mreži Internet [5]. U UDDI imenik usluga spremaju se informacije o programskim uslugama i njihovim pristupnim sučeljima. Kategorizacija spremljenih informacija najvažniji je aspekt UDDI specifikacije. Kategorizacija programskih usluga u UDDI imeniku usluga obavlja se poput kategorizacije u telefonskom imeniku gdje su informacije podijeljene na *bijele*, *žute* i *zelene* stranice. *Bijele* stranice sadrže informacije o pružateljima usluga. *Žute* stranice ostvaruju kategorizaciju programskih usluga prema funkcionalnosti koju usluge pružaju. *Zelene* stranice sadrže tehničke informacije potrebne za pristup i korištenje programskih usluga poput adrese na kojoj se nalazi WSDL opis pristupnog sučelja.

2.3.5 Proširenja Web Services tehnologije

Tehnologije SOAP, WSDL i UDDI su temeljne tehnologije potrebne za razvoj sustava zasnovanih na uslugama s osnovnim funkcionalnostima. Prošireni skup tehnologija WS-* sadrži dodatne tehnologije koje definiraju napredne mehanizme kao što su transakcije, orkestracija programskih usluga te sigurna i pouzdana komunikacija [6]. Slika 2.6 prikazuje prošireni stog WS-* skupa tehnologija.



Slika 2.6: Stog WS-* skupa tehnologija

Komunikacija programskih usluga zasniva se na razmjeni poruka stoga osnovu WS-* skupa tehnologija predstavljaju transportni protokoli. Za prijenos poruka mogu se koristiti različiti otvoreni protokoli poput protokola HTTP, SMTP i TCP, ali i različiti vlasnički (engl. *proprietary*) protokoli. Sloj za razmjenu poruka sadrži temeljne tehnologije koje definiraju protokole i mehanizme za razmjenu poruka. SOAP i WS-Addressing tehnologije temelje se na XML jeziku i stoga su neovisne o programskim tehnologijama, ali također postoje i tehnologije koje su specifične za određene programske tehnologije poput JMS (engl. *Java Message Service*). Sloj za opisivanje usluga sastoji se od WSDL i WS-Policy* tehnologija koje omogućuju razvijateljima

usluga detaljno opisivanje pristupnih sučelja programskih usluga. WSDL omogućuje opisivanje pristupnog sučelja programske usluge. WS-Policy* omogućuje opisivanje uvjeta korištenja programske usluge i naprednih funkcionalnosti koje usluga izlaže i/ili koristi. Sloj kvalitete usluge sadrži tehnologije koje omogućavaju izvršavanje transakcija (WS-AtomicTransaction, WS-BusinessActivity), pouzdanu razmjenu poruka (WS-ReliableMessaging) i sigurnu komunikaciju (WS-Security*). Sloj za kompoziciju usluga sadrži tehnologije WS-BPEL (engl. *Web Services Business Process Execution Language*) i WS-CDL (engl. *Web Service Choreography Description Language*) koje omogućuju izgradnju novih programskih usluga kompozicijom već postojećih uz korištenje mehanizama za koordinaciju programskih usluga i očuvanje konteksta poziva programskih usluga. Sloj za objavljivanje i otkrivanje sadrži tehnologije koje definiraju mehanizme za objavljivanje i razmjenu podataka o programskim uslugama na temelju kojih se izvršava odabir i pozivanje izabrane programske usluge. Važnija tijela zadužena za standardizaciju i definiranje WS-* stoga tehnologije su WS-I (engl. *Web Services Interoperability*), OASIS (engl. *Organisation for the Advancement of Structured Information Standards*), W3C (engl. *World Wide Web Consortium*) i IETF (engl. *Internet Engineering Task Force*).

3 Ispitivanje radnih značajki računalnih sustava

Razvoj računalnih mreža omogućio je oblikovanje i izgradnju raspodijeljenih računalnih sustava širokih namjena. Raspodijeljeni računalni sustavi pružaju korisnicima različite funkcionalnosti ovisno o namjeni raspodijeljenog računalnog sustava. Funkcionalnosti koje pružaju raspodijeljeni računalni sustav primjenjuju se u raznovrsnim domenama ljudskih djelatnosti. U znanosti i industriji postoji potreba za provođenjem složenih proračuna, obradom velikih količina podataka i izvođenjem složenih simulacija. U privredi postoji potreba za elektroničkim objedinjavanjem poslovanja različitih tvrtki i mogućnosti donošenja pravovremenih odluka i reakcija s obzirom na brze promjene tržišta. Krajnji korisnici koriste računalo većinu vremena za edukaciju, zabavu i dohvat informacija putem globalne mreže Internet koja čini najveći raspodijeljeni računalni sustav. Neki od najčešće korištenih primjenskih programa od strane krajnjih korisnika su Internet preglednici (engl. *Internet browsers*) koji se koriste za dohvat i pregled WWW stranica. Drugi često korišteni primjenski programi su programi za trenutnu razmjenu tekstualnih poruka (engl. *instant messaging*), programi za razmjenu elektroničke pošte, programi za pristupanje interesnim grupama (engl. *newsgroups*) i programi za razmjenu datoteka (engl. *file sharing*).

Iako se raspodijeljeni sustavi razlikuju po funkcionalnosti koju pružaju svi oni moraju pružati svoje funkcionalnosti uz određenu kvalitetu usluge (engl. *Quality of Service, QoS*). Ispitivanje, nadzor i održavanje kvalitete usluge raspodijeljenog sustava složen je zadatak budući da se raspodijeljeni sustavi mogu izvoditi u raznorodnim okolinama na različitim sklopovskim i programskim platformama. Stoga se postupak određivanja kvalitete usluge, nadzora računalne infrastrukture i primjenskih sustava nastoji automatizirati i olakšati primjenom sustava za ispitivanje radnih značajki usluga.

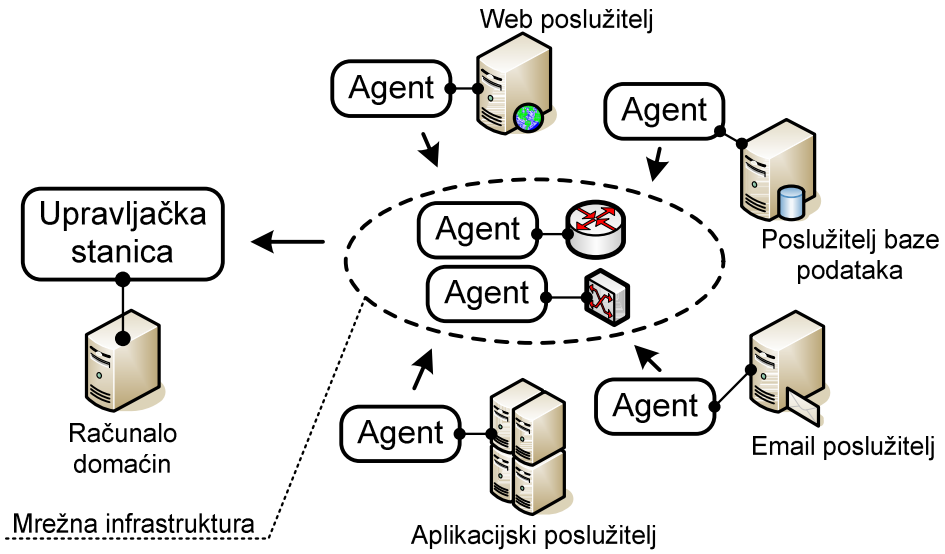
Zadaća sustava za ispitivanje radnih značajki je prikupljanje informacija o stanju računalne infrastrukture raspodijeljenog sustava, stanju mrežne infrastrukture i stanju programskih komponenti računalnog sustava. Nadalje, prikupljeni podaci obrađuju se i prikazuju administratoru sustava. Na temelju prikupljenih informacija administratori otkrivaju probleme unutar raspodijeljenog sustava, kao što su smanjena propusnost računalnih komponenti (engl. *bottleneck*s) i kvarovi računalne

infrastrukture. Nadalje, ovisno o vrsti otkrivenih problema administratori mogu donositi odluke o uvođenju promjena u postojeći raspodijeljeni sustav s namjerom poboljšavanja radnih značajki dostupnosti i pouzdanosti raspodijeljenog sustava.

U odjeljku 3.1 opisane su arhitekture dostupnih sustava za ispitivanje radnih značajki, a u odjeljcima 3.2, 3.3 i 3.4 prikazan je pregled trenutno dostupnih sustava na tržištu.

3.1 Arhitekture sustava za ispitivanje radnih značajki

Postoje tri arhitekture sustava za ispitivanje radnih značajki: arhitektura zasnovana na agentima (engl. *agent-based architecture*), arhitektura koje nije zasnovana na agentima (engl. *agentless architecture*) i hibridna arhitektura koja je zasnovana na značajkama prve dvije arhitekture. Osnovna razlika između arhitektura koje su zasnovane na agentima i arhitektura koje nisu zasnovane na agentima je u lokaciji podsustava zaduženog za prikupljanje informacija o računalnoj infrastrukturi, mrežnim elementima i primjenskim sustavima. Hibridna arhitektura je spoj prve dvije arhitekture i omogućava istovremeno prikupljanje radnih značajki sa i bez agenata.



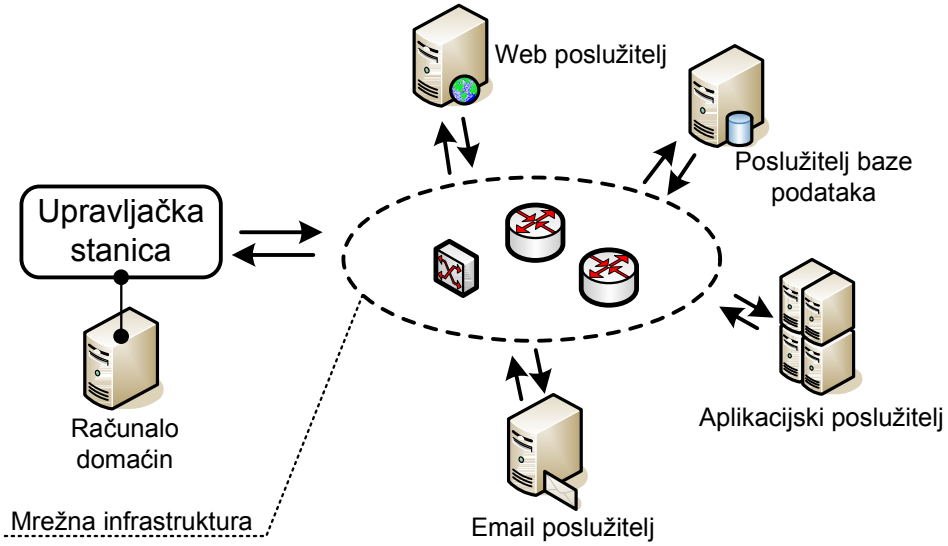
Slika 3.1: Arhitektura sustava zasnovanog na agentima

Arhitektura zasnovana na agentima zasnovana je na postavljanju agenata na sve elemente raspodijeljenog računalnog sustava čije radne značajke želimo očitavati. Slika 3.1 prikazuje arhitekturu sustava zasnovanu na agentima. Agent se definira kao programski kod koji obavlja dobro definirani zadatak na stroju domaćinu umjesto korisnika. Agente postavljaju administratori sustava ručno ili udaljeno preko

sučelja upravljačke stanice (engl. *management station*) ovisno o mogućnostima sustava za ispitivanje radnih značajki. Nakon postavljanja i podešavanja agenata sustav započinje s radom. Agenti ovisno o postavkama za rad očitavaju željene radne značajke, ako je potrebno obrađuje očitane radne značajke i stvaraju sažetu statistiku pogodnu za prijenos putem mrežne infrastrukture koju zatim šalju upravljačkoj stanici. Ako vrijednost određene radne značajke padne ispod dopuštene razine, agenti šalju upozorenja upravljačkoj stanici koja ih prosljeđuje administratoru. Za ovu arhitekturu karakteristično je potiskivanje (engl. *push*) radnih značajki, iako većina sustava koji omogućuju veću fleksibilnost podržava i istiskivanje (engl. *pull*) radnih značajki omogućujući dohvat radnih značajki na upit administratora ili same upravljačke stanice ovisno o njezinim radnim postavkama. Dohvaćene radne značajke prikazuju se administratorima primjenom preglednih vizualizacijskih tehnika. Prednost primjene agenata jest njihova mogućnost prikupljanja detaljnih informacija o stanju poslužitelja, primjenskih programa i mrežne infrastrukture, budući da su smješteni izravno na elementima na kojima vrše očitavanje radnih značajki. Prikupljanje radnih značajki u slučaju kvara mrežne infrastrukture također predstavlja prednost budući da za vrijeme kvara mrežne infrastrukture agenti mogu pohranjivati očitane radne značajke lokalno i prenijeti ih upravljačkoj stanici nakon oporavka mrežne infrastrukture. Nedostaci primjene agenata jesu postupak ručnog postavljanja i održavanja agenata, dodatno opterećenje koje agenti stvaraju na poslužiteljima, mrežnim elementima i primjenskim programima te nerijetko visoke cijene sustava za ispitivanje radnih značajki zasnovanih na agentima.

Arhitektura koja nije zasnovana na agentima temelji se na pretpostavci o dostupnosti radnih značajki putem standardima definiranih udaljenih sučelja (engl. *remote interfaces*) kojima se pristupa primjenom standardiziranih programskih tehnologija. Slika 3.2 prikazuje arhitekturu sustava koji nije zasnovan na agentima. Upravljačka stranica na zahtjev administratora ili u ovisnosti o radnim postavkama izvršava propitkivanje svih elemenata raspodijeljenog sustava i pribavlja željene radne značajke te ih prikazuje korisniku. Za ovu arhitekturu karakteristično je istiskivanje radnih značajki. Za pristup elementima raspodijeljenog računalnog sustava koriste se međuostalima programske tehnologije Telnet i SSH (engl. *Secure Shell*) za spajanje na UNIX platforme, a WMI (engl. *Windows Management Instrumentation*) za spajanje na Microsoft platforme. Prednosti primjene arhitekture

koja nije zasnovana na agentima jest jednostavnost postavljanja i održavanja sustava te često jeftinija nabavna cijena u odnosu na sustave zasnovane na agentima. Nedostaci su gubljenje mjernih značajki u slučaju kvara mrežne



Slika 3.2: Arhitektura sustava koji nisu zasnovani na agentima

infrastrukture, opterećenje mrežne infrastrukture budući da se prenose izravno očitane radne značajke bez prethodne obrade te nemogućnost očitavanja radnih značajki svih elemenata raspodijeljenog sustava zbog nepostojanja udaljenih sučelja za pristup elementima.

Sustavi za ispitivanje radnih značajki s hibridnom arhitekturom podržavaju istovremeno prikupljanje radnih značajki sa i bez agenata omogućujući veću fleksibilnost i raspoloživost sustava. Primjena hibridne arhitekture ima sve prednosti sustava koji prikupljaju radne značajke sa i bez agenata, a nedostaci primjene su velika nabavna cijena te složenost rada s takvim sustavom.

3.2 RSP

RSP sustav za ispitivanje radnih značajki razvila je Draconis Software programska kuća [7]. Kao i većina sustava za ispitivanje radnih značajki RSP sustav omogućuje ispitivanje radnih značajki i izvještavanje o stanjima nadziranih sustava. RSP sustav ima arhitekturu zasnovanu na agentima sa središnjim RSP poslužiteljem. Administratori postavljaju agente na sva računalu i poslužitelje čije se radne značajke žele mjeriti. Središnji RSP poslužitelj propitkuje (engl. *poll*) RSP agente i dohvaća mjerene radne značajke. Administratori i korisnici pristupaju mjerenim radnim

značajkama preko Web sučelja RSP poslužitelja. Središnji RSP poslužitelj i agenti komuniciraju putem globalne mreže Internet.

RSP agenti koriste module za prikupljanje radnih značajki računala na kojima se nalaze. Svaki modul implementira unaprijed definirano sučelje. Upotreba sučelja u komunikaciji s modulima omogućuje korisnicima izgradnju proizvoljnih (engl. *custom*) modula. Za izgradnju proizvoljnih modula korisnicima se nalaze na raspolaganju programski jezici C, C++, Perl i Java. Uz mogućnost izgradnje proizvoljnih modula RSP se isporučuje sa standardnim modulima. Moduli koji se nalaze na raspolaganju očitavaju iskorištenje memorije, procesora i diskovnog prostora, mrežnu statistiku, statistiku baza podataka, dostupnost usluga i mrežnih čvorova. Upotrebom sučelja modula mogućnosti proširivanja su velike omogućujući tako izgradnju modula za vlasničke (engl. *proprietary*) programske sustave. Dodavanjem i uklanjanjem modula RSP nadgleda samo željene radne značajke smanjujući pritom opterećenje računala domaćina. Slika 3.3 prikazuje logičku arhitekturu RSP agenta. Na osnovnoj

Slika 3.3: Logička arhitektura RSP agenta

razini moduli prikupljaju radne značajke. Prikupljene radne značajke pohranjuju se u bazu podataka na određeni vremenski period koji administrator definira. Nakon pohrane, poslovna logika trenutne radne značajke uspoređuje s postavljenim pragovima (engl. *tresholds*). Pragove postavljaju administratori putem Web sučelja. Ako trenutne radne značajke prelaze vrijednosti definirane pragovima poslovna logika šalje upozorenje administratorima stvaranjem kartice (engl. *ticket*). Svaka kartica sadrži detalje o poslužitelju, status i opis problema. Nakon rješavanja pojedinih problema karticama se dodaje opis rješenja problema te se kartice pohranjuju u bazu podataka kartica. Pritom se stvara arhiva koju je moguće

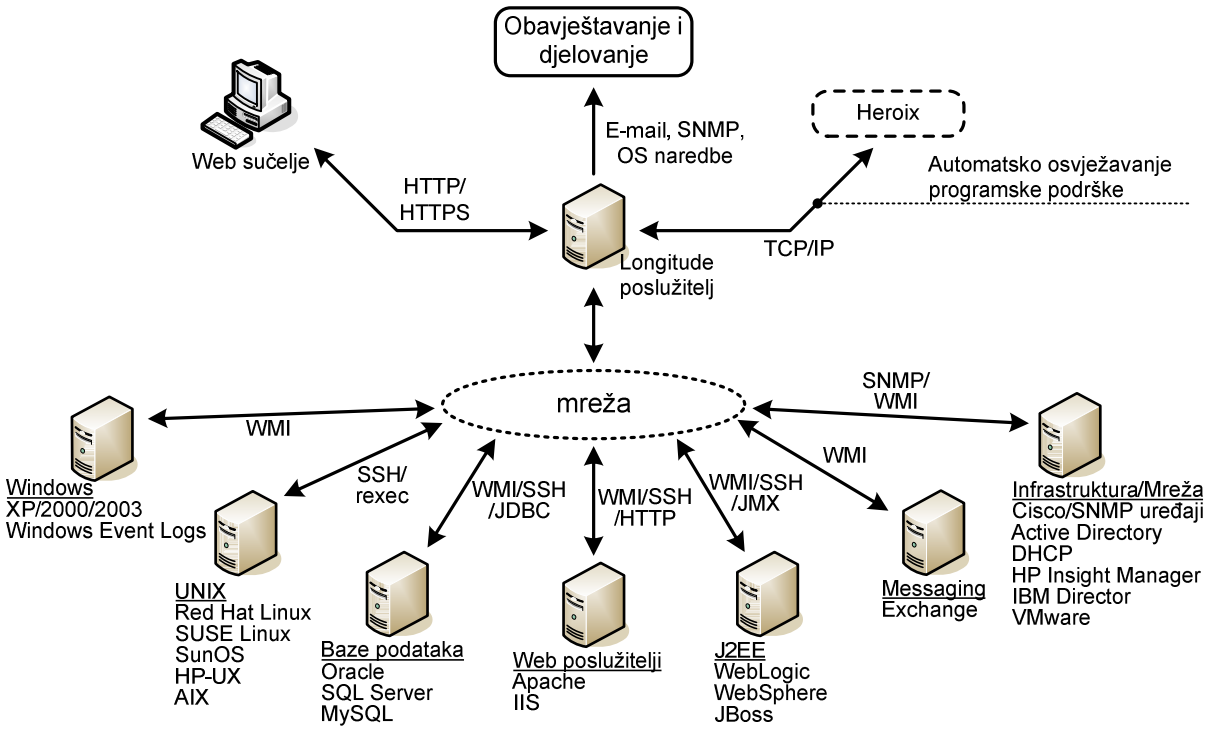
pretraživati radi pronalaska rješenja problema koji se iznova pojave. Radne značajke pohranjene u bazi podataka koristi središnji RSP poslužitelj za grafički prikaz korisnicima putem Web sučelja omogućujući analizu radnih značajki.

RSP sustav rješava karakterističnе probleme sustava zasnovanih na agentima, veliki utrošak vremena na ručno postavljanje i podešavanje agenata na računala i poslužitelje te osjetljivost sustava s obzirom na kvar središnjeg poslužitelja. RSP sustav omogućuje administratorima postavljanje agenata upotrebom instalacijski skripti i podešavanje radnih postavki agenata preko Web sučelja. Osjetljivost sustava s obzirom na kvar središnjeg poslužitelj smanjena je budući da RSP agenti imaju mogućnost samostalnog slanja upozorenja administratorima kada trenutne radne značajke premaše zadani prag.

3.3 Heroix Longitude

Heroix Longitude je sustav za ispitivanje radnih značajki, arhitekture koja nije zasnovana na agentima, koji omogućava ispitivanje radnih značajki i izvještavanje o stanjima sustava koji se nadgledaju [8]. Ispitivanje dostupnosti i radnih značajki moguće je provoditi na web poslužiteljima, aplikacijskim poslužiteljima, bazama podataka, poslužiteljima elektronske pošte, J2EE (engl. *Java 2 Platform, Enterprise Edition*), mrežnim uređajima, virtualnim strojevima (engl. *virtual machines*) i Windows, Linux i UNIX okolinama.

Slika 3.4 prikazuje arhitekturu sustava Heroix Longitude. Administrator postavlja Longitude poslužitelj primjenom jednostavnog i brzog instalacijskog postupka. Nakon postavljanja, Longitude poslužitelj automatski otkriva (engl. *auto discovery*) sve entitete pogodne za ispitivanje radnih značajki oslobađajući administratora postupka ručnog unošenja. Nakon otkrivanja, zadaju se radne postavke putem Web sučelja, koje se prenose do Longitude poslužitelja i sustav zatim započinje s ispitivanjem radnih značajki. Za ispitivanje radnih značajki onih entiteta koji to omogućuju Longitude poslužitelj koristi programske tehnologije WMI, SSH, HTTP, JDBC (engl. *Java Database Connectivity*), JMX (engl. *Java Management Extensions*) i SNMP (engl. *Simple Network Management Protocol*). WMI sučelje omogućuje pristupanje Microsoft Windows operacijskim sustavima, prikupljanje radnih značajki i upravljanje izvođenjem odgovarajućih skripti na udaljenim računalima. SSH je mrežni protokol za spajanje i izvođenje naredbi na



Slika 3.4: Arhitektura sustava Heroix Longitude

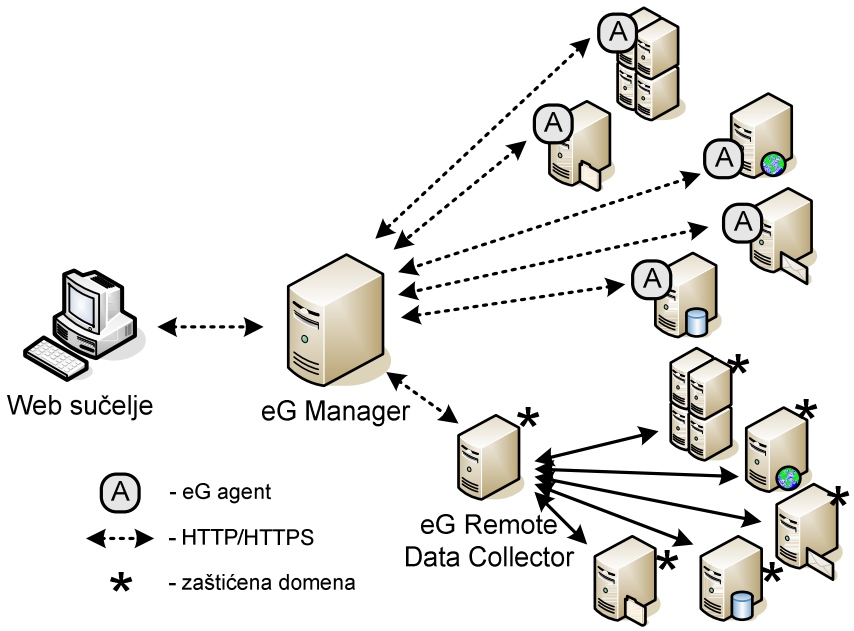
udaljenom računalu preko sigurnog kanala. JDBC je JAVA programska tehnologija za spajanje na relacijske baze podataka i izvršavanje raznih upita. JMX je JAVA programska tehnologija koja pruža mogućnosti udaljenog upravljanja i nadzora primjenskih sustava i sistemskih uređaja. SNMP je skup standarda koji omogućuju upravljanje i nadgledanje mrežnih uređaja. Primjena svih tih programskih tehnologija ne zahtjeva posebno postavljanje programskog koda na entitete koji se žele nadgledati, budući da većina entiteta ima ugrađenu podršku za neku od navedenih programskih tehnologija.

Heroix Longitude sustav ima određene pogodnosti koje olakšavaju analizu i upravljanje radnim značajkama. U radu s velikim brojem entiteta kojima se ispituju radne značajke sustav omogućava logičko grupiranje entiteta ovisno o lokaciji, odjelu kojemu pripadaju, operacijskom sustavu, administratoru i vremenskoj zoni olakšavajući detaljno pregledavanje željenih skupina entiteta i njihovih mjerenih radnih značajki. Nakon samog postavljanja Longitude poslužitelj ima definirane pragove za sve radne značajki na temelju kojih se u slučaju kada vrijednost radne značajke padne ispod ili naraste iznad definirane vrijednosti praga stvaraju upozorenja koja se prosljeđuju administratorima putem elektroničke pošte. Administrator može definirati korelirane događaje (engl. *correlated events*) na temelju

kojih se okidaju (engl. *trigger*) složena pravila koje je administrator definirao putem Web sučelja. Složena pravila mogu uključivati slanje obavijesti i upozorenja, ali i izvođenje korektivnih naredbi na lokalnim i udaljenim sustavima. Korelirani događaji su događaji koji obuhvaćaju više uvjeta o stanju ispitivanih entiteta koji utječu na poslovne procese ili razinu kvalitete usluge. Za potrebe ispitivanja kvalitete pruženih usluga Longitude koristi umjetne web transakcije (engl. *synthetic web transactions*). Primjenom umjetnih web transakcija Longitude sustav oponaša krajnjeg korisnika, stvara vezu s pruženim uslugama, opterećuje ih i pritom mjeri kvalitetu pruženih usluga. Sve prikupljene radne značajke pohranjuju se u bazu podataka, na temelju kojih se izrađuju detaljni grafički prikazi radnih značajki poput minimalne, maksimalne i prosječne vrijednosti.

3.4 eG Enteprise Suite

Sustav eG Enterprise Suite je skup programskih rješenja za prikupljanje, obradu i vizualizaciju radnih značajki cjelokupne računalne i mrežne infrastrukture, primjenskih programa i usluga te predstavlja jedno od najpotpunijih rješenja dostupnih na tržištu [9]. Sustav je zasnovan na hibridnoj arhitekturi koja omogućuje veću fleksibilnost i prilagodljivost i ispitivanje radnih značajki gotovo svih entiteta koji se mogu nadzirati. Arhitektura sustava eG Enterprise Suite sustava prikazana je na slici 3.5.



Slika 3.5: Arhitektura eG Enterprise Suite sustava

Središnja komponenta sustava je eG Manager koja je zadužena za središnje podešavanje i upravljanje eG agentima. Administratori pristupaju prikupljenim radnim značajkama preko središnjeg Web portala smještenog na komponenti eG Manager koja omogućuje autentifikaciju administratora. Komponenta eG Manager komunicira s eG agentima upotrebom protokola HTTP ili HTTPS. Komunikacija između eG agenta i komponente eG Manager, tijekom prikupljanja radnih značajki, jednosmjerna je i inicirana od strane eG agenta. Nakon postavljanja, eG agenti očitavaju radne značajke u skladu s radnim postavkama i zatim ih obrađuju smanjujući tako opterećenje na komponenti eG Manager koja primljene radne značajke pohranjuje u bazu podataka. Za očitavanje radnih značajki eG agenti koriste programske tehnologije ICMP (engl. *Internet Control Message Protocol*), SNMP, JMX, WMI/PerfMon, ISAPI (engl. *Internet Server Application Programming Interface*) i SQL (engl. *Structured Query Language*) i dr. Postoje dvije vrste eG agenata vanjski i unutarnji. Vanjski eG agenti imaju mogućnost prikupljanja radnih značajki s onih mrežnih elemenata koji imaju sučelja koja omogućuju udaljeni pristup. Unutarnji eG agenti koriste se kada se radne značajke ne mogu prikupiti udaljeno ili se zahtjeva veća točnost i preciznost radnih značajki i tada je potrebno postaviti agenta na sustav čije radne značajke želimo mjeriti. Uz podršku prikupljanju radnih značajki primjenom agenata, sustav eG Enterprise Suite omogućuje i prikupljanje radnih značajki koje nije zasnovano na agentima. Osnovu prikupljanju radnih značajki bez agenata pruža komponenta eG Remote Data Collector koja ima dozvole za pristup svim entitetima koji imaju u sebi ugrađene mehanizme za mjerenje radnih značajki. Prikupljene radne značajke eG Remote Data Collector dostavlja komponenti eG Manager primjenom protokola HTTP ili HTTPS. Prikupljanje radnih značajki eG Remote Dana Collector ostvaruje se primjenom različitih protokola i mehanizama. WMI/Perfmon koristi se za prikupljanje radnih značajki Microsoft Windows operacijskih sustava i Microsoft primjenskih programa, JMX/HTTP koristi se za prikupljanje radnih značajki WebLogic primjenskog programa, SQLNet za Oracle baze podataka, SSH za operacijske sustave Solaris, Linux, AIX i HPUX.

Iako sustav eG Enterprise Suite omogućava nadzor velikog broja entiteta, uvijek postoje entiteti koji nisu podržani. Stoga sustav eG Enterprise Suite omogućuje izgradnju proizvoljnih nadzornih funkcija za vlasničke sustave i primjenske programe, koji nemaju ugrađenu podršku za mjerenje radnih značajki, preko Web sučelja

komponente eG Manager. Web sučelje omogućuje izgradnju proizvoljnih nadzornih funkcija, ali i složenih testova primjenskih programa. Komponenta koja omogućuje proširivanje zove se Integration Console i ona je proširenje komponente eG Manager.

Neke od prednosti korištenja sustava eG Enterprise Suite su automatsko generiranje vrijednosti pragova na temelju analize radnih značajki pohranjenih u bazi podataka, obavještavanje administratora kada je to potrebno primjenom SMS (engl. *Short Message Service*) poruka i elektroničke pošte, automatska trijaža (engl. *triage*) problema pri kojoj se odvajaju uzroci i posljedice. Primjena Web portala i personalizacija Web sučelja omogućuju izgradnju zasebnih Web sučelja za korisnike ovisno o njihovim ulogama. Izgradnjom zasebnog Web sučelja moguće je administratorima prikazivati samo ona sredstva za koja su zadužena, olakšavajući tako rad i povećavajući sigurnost sustava. Uz navedene prednosti, sustav eG Enterprise suite pruža još mnoge druge [10] što ga čini jednim od najpotpunijih rješenja za ispitivanje radnih značajki na tržištu.

4 Razvojna okolina PIE

Razvoj raspodijeljenih sustava zahtjeva poznavanje velikog broja standarda, korištenje različitih programskih tehnologija i razvojnih okolina. Tijekom procesa razvoja raspodijeljenih sustava razvijatelji sustava utroše značajan udio vremena za razvoj općenitih mehanizama koji se pojavljuju u sustavima zasnovanim na uslugama. S ciljem omogućavanja lakšeg i bržeg razvoja primjenskih raspodijeljenih sustava zasnovanih na uslugama izgrađena je razvojna okolina PIE (engl. *Programmable Internet Environment*).

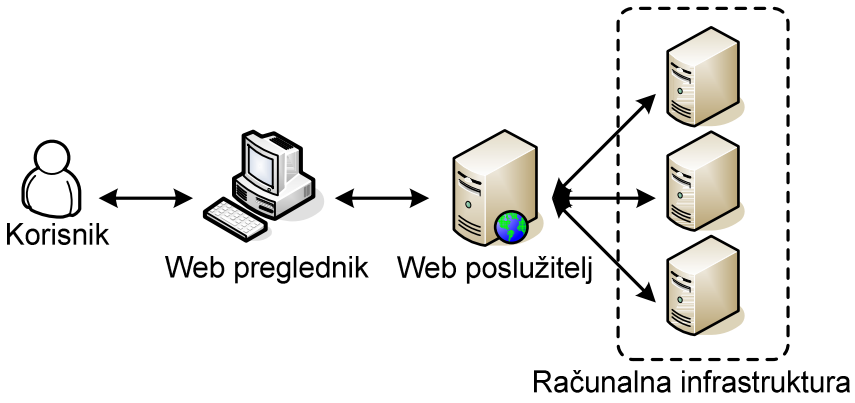
Razvojnu okolinu PIE izgradila je istraživačka grupa RIS (engl. *Research on Intranet/Internet Systems*) okupljena na Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave na Fakultetu elektrotehnike i računarstva. PIE razvojna okolina ostvarena je u okviru podprojekta "*Okrilje posrednika javnog informacijskog sustava"* nacionalnog tehnologijskog poliprojekta CRO-GRID uz suradnju tvrtke Ericcson Nikola Tesla i uz potporu Ministarstva znanosti, obrazovanja i športa. Izgrađena razvojna okolina sastoji se od Web korisničkog sučelja, Sustava prividne mreže, Sustava za upravljanje uslugama, Koopeticijskih mehanizama, Sustava za izgradnju raspodijeljenih programa, Sustava za izvođenje raspodijeljenih programa i Sustava za nadzor pristupa.

Web korisničko sučelje omogućuje upravljanje čvorovima prividne mreže, postavljanje programskih usluga te razvoj raspodijeljenih programa. Sustav prividne mreže omogućuje učinkovito upravljanje računalnom infrastrukturom. Sustav za upravljanje uslugama omogućuje automatizirano postavljanje i otkrivanje programskih usluga. Koopeticijski mehanizmi u suradnji sa Sustavom za izgradnju i Sustavom za izvođenje raspodijeljenih programa pružaju potporu izgradnji i izvođenju raspodijeljenih programa. Sustav za nadzor pristupa omogućuje izgradnju sigurne prividne raspodijeljene računalne okoline.

U odjeljku 4.1 opisano je Web korisničko sučelje, a u odjeljku 4.2 Sustav prividne mreže. Sustav za upravljanje uslugama opisan je u odjeljku 4.3. Sustavi koji pružaju potporu izgradnji i izvođenju raspodijeljenih programa opisani su u odjeljcima 4.4, 4.5 i 4.6. Odjeljak 4.7 opisuje Sustav za nadzor pristupa.

4.1 Upravljanje razvojnom okolinom PIE

Upravljanje razvojnom okolinom PIE omogućuje Web korisničko sučelje Sustava za upravljanje i nadzor. Putem Web korisničkog sučelja moguće je dodavati i uklanjati čvorove prividne mreže, postavljati programske usluge na čvorove prividne mreže, postavljati komunikacijske i sinkronizacijske mehanizme, postavljati infrastrukturu za izvođenje raspodijeljenih programa i razvijati raspodijeljene programe.



Slika 4.1: Arhitektura Sustava za upravljanje i nadgledanje

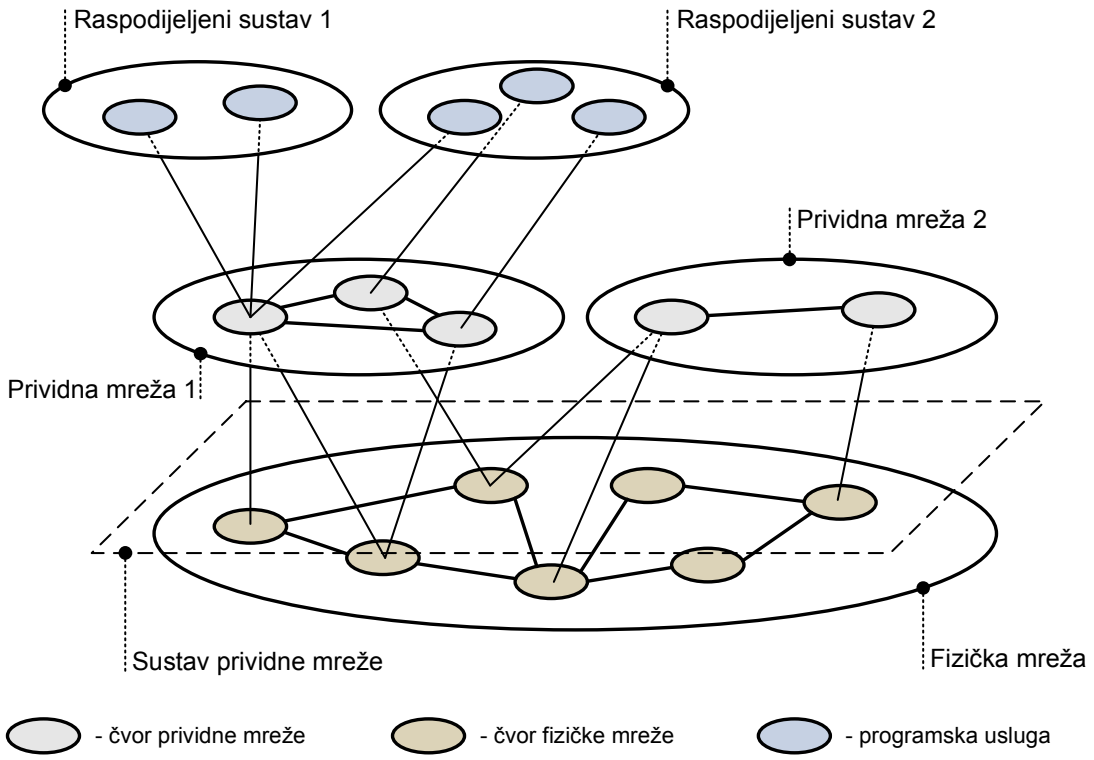
Slika 4.1 prikazuje arhitekturu Sustava za upravljanje i nadgledanje. Korisnik primjenom bilo kojeg Web preglednika pristupa Web korisničkom sučelju. Putem Web korisničkog sučelja korisnik izvršava akcije koje uzrokuju slanje odgovarajućih zahtjeva Web poslužitelju. Web poslužitelj primljene zahtjeve prosljeđuje računalnoj infrastrukturi koja izvršava korisnikove zahtjeve. Osim što prosljeđuje zahtjeve Sustav za upravljanje i nadzor izvršava nadzor računalne infrastrukture i prikuplja informacije o stanju sustava koje zatim prosljeđuje Web pregledniku na prikaz korisniku.

Web korisničko sučelje izrađeno je primjenom *ASP .NET* tehnologije koja je dio Microsoft *.NET* radnog okvira. Pristup Web korisničkom sučelju ostvaren je primjenom IIS (engl. *Internet Information Services*) poslužitelja. Poslovna logika Sustava za upravljanje i nadzor ostvarena je primjenom C# programskog jezika.

4.2 Sustav prividne mreže

Sustav prividne mreže (engl. *Overlay*) omogućuje učinkovito upravljanje računalnom infrastrukturom prilikom oblikovanja, izgradnje i izvođenja raspodijeljenih primjenskih programa [11]. Prividna mreža čini temeljni gradivni blok programske

okoline za izgradnju raspodijeljenih sustava koji omogućuje dinamičko dodavanje i uklanjanje čvorova prividne mreže, pokretljivost čvorova prividne mreže, veću prilagodljivost raspodijeljenih sustava te povećanu sigurnost i pouzdanost komunikacije. Primjenom prividne mreže gradi se prividna raspodijeljena programska okolina.



Slika 4.2: Prividna raspodijeljena okolina

Slika 4.2 prikazuju prividnu raspodijeljenu programsku okolinu. Osnovu komunikacijske infrastrukture prividne raspodijeljene programske okoline čini fizička mreža. Unutar fizičke mreže nalazi se konačan broj čvorova od kojih je svakom dodijeljena fizička adresa. Povrh fizičke mreže gradi se proizvoljan broj prividnih mreža od kojih svaka sadrži svoj logički komunikacijski prostor. Unutar prividne mreže komunikacija se zasniva na upotrebi logičkih adresa. Primjenom Sustava prividne mreže provodi se virtualizacija fizičke mreže. Povrh prividne mreže grade se raspodijeljeni sustavi zasnovani na uslugama koji se sastoje od programskih usluga koje međusobno komuniciraju unutar logičkog komunikacijskog prostora.

Razvojem raspodijeljenih sustava povrh globalne mreže Internet grade se raspodijeljeni sustavi s izvjesnim ograničenjima koja proizlaze iz mehanizama

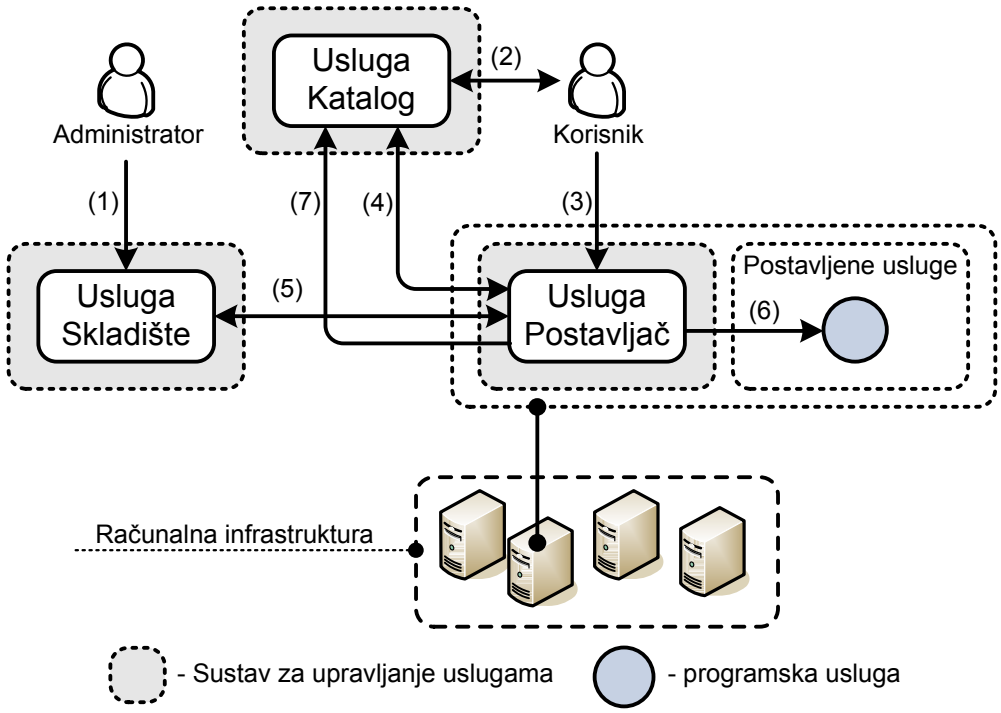
adresiranja i komunikacije same globalne mreže Internet. Svaki čvor koji je dio globalne mreže Internet jedinstveno je određen svojom IP (engl. *Internet Protocol*) ili DNS (engl. *Domain Name System*) adresom koja je najčešće fiksna. Pristupanje postavljenim uslugama na čvorovima globalne mreže Internet moguće je jedino uz poznavanje fizičke adrese čvora na kojem je usluga postavljena. Iz ovoga slijedi da fizički komunikacijski prostor čvrsto povezuje uslugu s mjestom njezina postavljanja u globalnoj mreži Internet. Premještanjem usluga s jednog računala na drugo ili kvarom računala na kojem su postavljanje usluge onemogućen je ispravan rad raspodijeljenog sustava zasnovanog na uslugama izgrađenog povrh globalne mreže Internet što zahtijeva unošenje promjena u izgrađeni raspodijeljeni sustav.

Sustav prividne mreže navedeno ograničenje rješava stvaranjem prividne mreže (engl. *Virtual Network*) i logičkog komunikacijskog prostora. Logički komunikacijski prostor stvara se povrh fizičkog komunikacijskog prostora, a prividna mreža povrh fizičke mreže. Svakom čvoru prividne mreže dodjeljuje se jedna ili više logičkih adresa koje su neovisne o fizičkim adresama. Prijenos podataka između čvorova prividne mreže omogućuje Sustav prividne mreže koji logičke adrese preslikava u fizičke i obrnuto omogućujući tako prijenos poruka primjenom fizičke mreže. Postavljanjem usluga na čvorove prividne mreže izvršava se čvrsto povezivanje usluga s logičkim adresama čvorova na koje se postavljaju. Ispadom fizičkog čvora nije potrebno unositi promjene u raspodijeljeni sustav, nego je potrebno premjestiti postavljene usluge na drugo računalo i tom računalu dodijeliti logičke adrese računala koje je nedostupno. Uz veću pouzdanost raspodijeljenih sustava primjena Sustava prividne mreže omogućuje bržu i efikasniju komunikaciju unutar prividne mreže izgradnjom vlastitih Podsustava za usmjeravanje poruka i razrješavanje adresa unutar prividne mreže. Podsustavi za usmjeravanje sadrže tablice usmjeravanja pomoću kojih se na osnovu logičke adrese primatelja dinamički određuje logička adresa čvora kojem je potrebno proslijediti poruku. Poruka se može usmjeravati od izvorišta do odredišta preko više logičkih čvorova. Podsustavi za razrješavanje adresa sadrže tablice adresa, koje preslikavaju logičke adrese u fizičke, pomoću kojih se na temelju logičke adrese određuje fizička adresa čvora kojem je poruku potrebno proslijediti. Upotreba Sustava prividne mreže pridonosi sigurnosti raspodijeljenih sustava koji su izgrađeni povrh prividne mreže budući da računala koja nisu dio prividne mreže ne mogu narušiti njihov rad.

4.3 Sustav za upravljanje uslugama

Sustav za upravljanje uslugama omogućuje automatizirano postavljanje i otkrivanje programskih usluga u prividnoj raspodijeljenoj računalnoj okolini [12]. Vlasnik usluge gradi instalacijski paket koji sadrže sve potrebne podatke za postavljanje i korištenje programske usluge. Na temelju izgrađenog instalacijskog paketa programsku uslugu moguće je postaviti primjenom Sustava za upravljanje usluga. Nakon postavljanja programskih usluga Sustav za upravljanje uslugama osvježava podatke o dostupnim uslugama i tako omogućuje otkrivanje i korištenje upravo postavljanje programske usluge. Sustav za upravljanje uslugama je raspodijeljeni sustav koji se sastoji od usluga Katalog, Skladište i Postavljač.

Usluga Katalog pruža potporu otkrivanju i održavanju informacija o postavljenim programskim uslugama. Usluga Skladište koristi se za spremanje instalacijskih paketa programskih usluga. Usluga Postavljač predstavlja središnju uslugu Sustava za upravljanje uslugama koja je zadužena je za postavljanje i uklanjanje programskih usluga te osvježavanje informacija pohranjenih u usluzi Katalog. Slika 4.3 prikazuje arhitekturu Sustava za upravljanje uslugama.



Slika 4.3: Arhitektura Sustava za upravljanje uslugama

Administrator gradi instalacijski paket programske usluge koji sprema u sustav (1) primjenom usluge Skladište. Korisnik pretražuje (2) imenik usluga primjenom

usluge Katalog i dohvaća informacije potrebne za pokretanje (3) postupka postavljanja programske usluge. Usluga Postavljač pribavlja (4) adresu lokacije na kojoj je spremljen instalacijski paket odgovarajuće programske usluge korištenjem usluge Katalog i zatim dohvaća (5) sam instalacijski paket programske usluge. Na temelju podataka pohranjenih u instalacijskom paketu usluga Postavljač lokalno postavlja (6) programsku uslugu. Nakon postavljanja programske usluge usluga Postavljač registrira (7) postavljenu uslugu u imenik usluga primjenom usluge Katalog.

Usluga Skladište omogućuje spremanje i uklanjanje instalacijskih paketa koje administratori sustava grade. Instalacijski paket sastoji se od arhive datoteka, instalacijske skripte i opisa sučelja programske usluge. Arhiva datoteka je arhiva izvršnih, konfiguracijskih i podatkovnih datoteka programske usluge. Instalacijska skripta je tekstualna datoteka koja opisuje instalacijski postupak programske usluge. Opis sučelja programske usluge je WSDL dokument koji opisuje pristupno sučelje programske usluge. Prilikom spremanja instalacijskog paketa usluga Skladište prosljeđuje adresu čvora prividne mreže, na kojem je postavljena, i ime instalacijskog paketa usluzi Katalog koja omogućuje otkrivanje usluga. U prividnoj raspodijeljenoj računalnoj okolini može se nalaziti više usluga Skladišta čime je povećana pouzdanost samog sustava. Usluga Katalog omogućuje otkrivanje i održavanje informacija o programskim uslugama. Usluga Katalog ostvarena je kompozicijom usluga. Sastoji se od jedne usluge Globalni Katalog i više usluga Lokalni Katalog. Usluga Lokalni Katalog postavlja se na svaki čvor prividne mreže i koristi se za spremanje informacija o lokalno postavljenim i instanciranim programskim uslugama. Usluga Globalni Katalog smješta se na jedan čvor prividne mreže čija je adresa poznata svima i koristi se za spremanje informacija o registraciji instalacijskih paketa i informacija o prividnoj računalnoj okolini kao što je popis adresa svih čvorova prividne mreže. Ovakvom arhitekturom usluge Katalog radna svojstva sustava ne ovise o količini spremljenih podataka. Usluga Postavljač omogućuje postavljanje, uklanjanje i konfiguriranje programskih usluga. Svaki čvor prividne mreže može imati postavljenu uslugu Postavljač. Usluga Postavljač na temelju informacija pribavljenih iz usluge Katalog pribavlja instalacijski paket iz usluge Skladište i postavlja programsku uslugu na čvoru domaćinu.

4.4 Koopeticijski mehanizmi

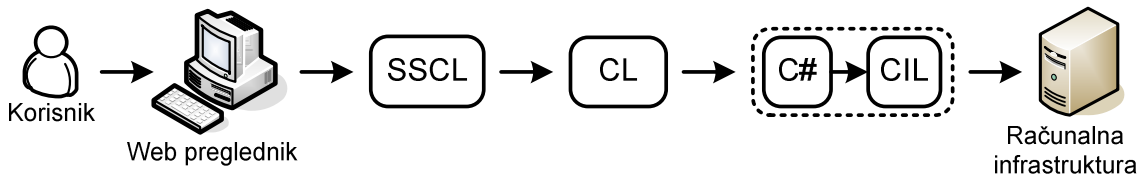
Koopeticijski mehanizmi koriste se za sinkronizaciju, komunikaciju, suradnju i natjecanje u prividnoj računalnoj okolini [13]. Naziv koopeticijski je složenica engleskih riječi za suradnju (engl. *cooperation*) i natjecanje (engl. *competition*). Tijekom razvoja raspodijeljenih sustava grade se raspodijeljeni programi koji upravljaju izvođenjem raspodijeljenog sustava. Raspodijeljeni programi surađuju razmjenom podataka primjenom koopeticijskih mehanizama, a natječu pri zauzimaju sredstava. Koopeticijski mehanizmi podržavaju dva mehanizma dohvata i zauzimanja sredstava zasnovana na propitkivanju (engl. *poll*) i povratnom pozivu (engl. *callback*). U oba slučaju ako je sredstvo raspoloživo ono se dodjeljuje. Ako se koristi propitkivanje i traženo sredstvo nije slobodno korisniku se vraća negativan odgovor i korisnik mora slati ponovno zahtjev. Ako se koristi povratni poziv i traženo sredstvo nije slobodno korisnikov zahtjev stavlja se u rep čekanja i korisnik je obaviješten o zauzeću sredstva. Kada korisnikov zahtjev dođe na početak repa čekanja korisniku se šalje poruka o oslobođenju sredstva i raspoloživo sredstvo.

Ostvareni koopeticijski mehanizmi su binarni semafor, opći semafor, poštanski pretinac i usmjernik događaja. Binarni i opći semafori omogućuju sinkronizaciju raspodijeljenih programa i zaštitu pristupa dijeljenim sredstvima. Raspodijeljeni program može pristupiti dijeljenom sredstvu samo ako je prethodno zauzeo semafor. Binarni semafor istovremeno može zauzeti samo jedan raspodijeljeni program, dok opći semafor istovremeno može zauzeti ograničen broj raspodijeljenih programa. Poštanski pretinac omogućuje komunikaciju raspodijeljenih programa razmjenom poruka. Poruke su XML dokumenti i stoga su neovisni o korištenim programskim tehnologija i računalnim platformama. Poštanski pretinac ima dvije podatkovne strukture rep poruka i rep zahtjeva. U rep zahtjeva spremaju se povratne adrese pozivatelja. Kad je rep zahtjeva prazan u rep poruka spremaju se pristigle poruke. Kad je rep zahtjeva neprazan prva pristigla poruka šalje se pozivatelju koji se nalazi na početku repa. Usmjernik događaja predstavlja mehanizam komunikacije objava/pretplata (engl. publish/subscribe) zasnovan na događajima. Usmjernik događaja sadrži skupove događaja i skupove pretplata. U skupove događaja spremaju se događaji objavljeni u usmjerniku događaja. U skup pretplata spremaju se sve zaprimljene pretplate. Primjena usmjernika događaja podrazumijeva postojanje objavitelja (engl. *publishers*), pretplatnika (engl. *subscribers*) i

interpretatora (engl. *interpreters*). Objavitelji objavljuju događaje u usmjerniku događaja. Pretplatnici se pretplaćuju na događaje spremljene u usmjerniku događaja. Interpretatori ostvaruju logiku usmjeravanja događaja prema odgovarajućim pretplatnicima.

4.5 Sustav za izgradnju raspodijeljenih programa

Sustav za izgradnju raspodijeljenih programa omogućuje izgradnju raspodijeljenih programa primjenom SSCL (engl. *Simple Service Composition Language*) programskog jezika [13]. Struktura SSCL programskog jezika slična je jednostavnim skriptnim jezicima. Budući da je SSCL programski jezik visoke razine izgradnji raspodijeljenih programa mogu pristupiti i krajnji korisnici s malim iskustvom u programiranju. Programi napisani u SSCL programskom jeziku prevode se u CL (engl. *Coopetition Language*) programski jezik. CL programski jezik omogućava izgradnju raspodijeljenih primjenskih programa primjenom hibridne metode kompozicije usluga koja se zasniva na primjeni metoda orkestracije i koreografije. Programski jezik CL zasnovan je na jeziku XML i koristi podskup naredbi programskog jezika BPEL4WS. Usklađenost CL programskog jezika s Web Services standardima omogućava korištenje raznovrsnih programskih sustava. Programi napisani CL programskim jezikom prevode se dvostupanjski u CIL (engl. *Common Intermediate Language*) programski jezik. U prvom stupnju CL programski jezik prevodi se u C# programski jezik koji je dio razvojnog okvira Microsoft *.NET*. Zatim se prevedeni program u C# programskom jeziku prevodi u jezik CIL. Prevedeni programi u jeziku CIL izvode se primjenom virtualnog stroja Microsoft *.NET* radnog okvira koji je postavljen na svim računalima računalne infrastrukture. Slika 4.4 prikazuje opisani postupak prevođenja programskih jezika.

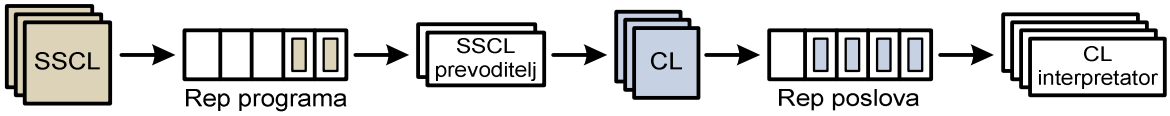


Slika 4.4: Hijerarhija programskih jezika

4.6 Sustav za izvođenje raspodijeljenih programa

Raspodijeljeni programi grade se primjenom primjenskih usluga, koopeticijskih mehanizama i raspodijeljenih programa. Primjenske usluge i koopeticijski mehanizmi

postavljaju se primjenom Sustava za upravljanje uslugama. Raspodijeljeni programi grade se primjenom SSCL programskog jezika. Arhitektura Sustava za izvođenje raspodijeljenih programa prikazana je na slici 4.5.



Slika 4.5: Arhitektura Sustava za izvođenje raspodijeljenih programa

Razvijatelj raspodijeljenih programa primjenom Web korisničkog sučelja i SSCL programskog jezika gradi raspodijeljeni program. Nakon izgradnje započinje proces prevođenja raspodijeljenih programa. Raspodijeljeni programi koji upravljaju tijekom izvođenja pojedinačno se spremaju u rep programa. Rep programa i rep poslova ostvareni su primjenom koopeticijskog mehanizma poštanski pretinac [14]. SSCL prevoditelji povezuju se na rep programa. SSCL prevoditelji koji su povezani na isti rep programa natječu se međusobno pri dohvatu i prevođenju raspodijeljenih programa. SSCL prevoditelji prevođenjem stvaraju raspodijeljene programe u CL programskom jeziku koje spremaju u rep poslova. CL interpretatori povezuju se na rep poslova. CL interpretatori povezani na isti rep poslova međusobno se natječu pri dohvatu, prevođenju i izvođenju raspodijeljenih programa iz repa poslova. Svaki CL interpretator dohvaća jedan CL raspodijeljeni program, prevodi ga i zatim izvodi. Izvođenje CL raspodijeljenog programa započinje prevođenjem CL raspodijeljenog programa u C# programski jezik, a potom u CIL programski jezik. Nakon prevođenja CL interpretator dobiveni programski kod postavlja u odgovarajuće kazalo i povezuje ga s Microsoft IIS i *ASP .NET* poslužiteljima i zatim ga pokreće. Po završetku izvođenja CL interpretator uklanja sve stvorene poveznice i programski kod s lokalnog računala.

4.7 Sustav za nadzor pristupa

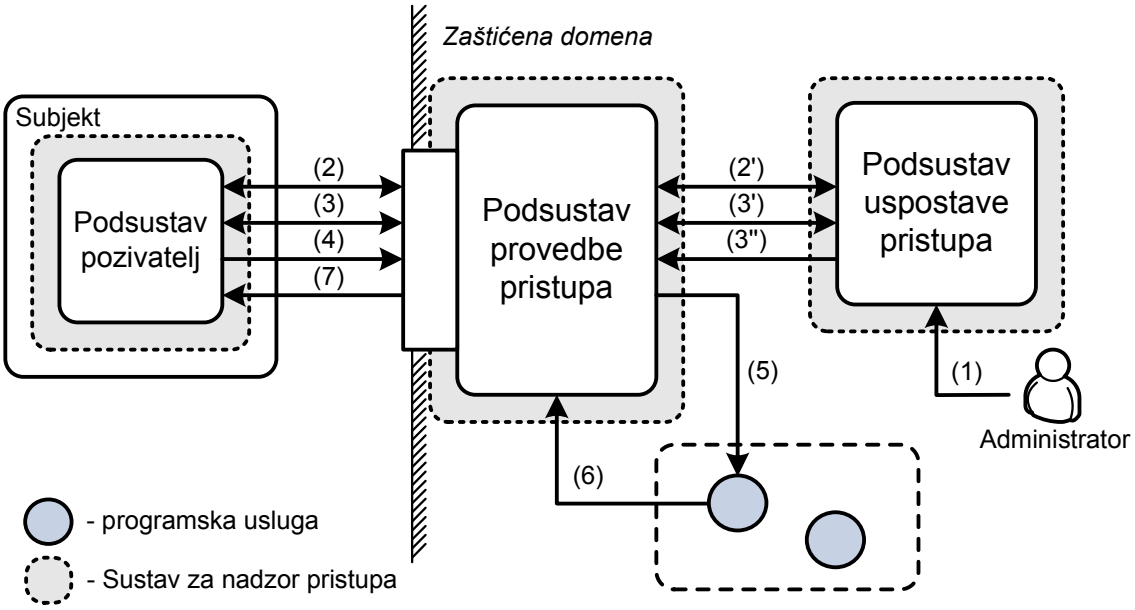
Sustav za nadzor pristupa omogućuje nadzor pristupa u prividnoj raspodijeljenoj okolini tako što ostvaruje funkcionalnost sigurnosnog posrednika u komunikaciji između korisnika i usluga [15]. Primjenom Sustava za nadzor pristupa grade se zaštićene domene nad skupovima usluga. Samo ovlašteni korisnici imaju pravo pristupa usluzi unutar zaštićene domeni. Sustav za nadzor pristupa omogućuje dodavanje i uklanjanje novih usluga i korisnika u zaštićenu domenu te prilagođavanje

nadzora pristupa zahtjevima usluge. Dodavanje i uklanjanje novih usluga u zaštićenu domenu ostvaruje se putem javno dostupnih postupaka registracije i autentifikacije. Prilikom dodavanja novih usluga u zaštićenu domenu pružatelji usluga definiraju sigurnosne zahtjeve usluga na osnovi kojih Sustav za nadzor pristupa donosi odluke koje mehanizme treba primijeniti tijekom nadzora pristupa usluzi. Postupak prijave usluge u novu domenu zasniva se na povjerenju pružatelja usluge u administratora sustava budući da na temelju njihovog dogovora administrator sustava postavlja prava pristupa uslugama zaštićene domene. Na temelju postavljenih prava pristupa Sustav za nadzor pristupa provodi postupak provjere autorizacije pristupa uslugama.

Sustav za nadzor pristupa tijekom rada koristi tri skupine podataka: svojstva identiteta, skup prava i skup odredbi. Svojstva identiteta sadrže podatke o korisnicima i usluga koje se koriste pri postupku autentifikacije. Podatke o uslugama i korisnicima unose pružatelji usluga odnosno korisnici tijekom registriranja na Sustav za nadzor pristupa. Skup prava definira administrator sustava u dogovoru s pružateljem usluge. Podaci skup prava koriste se pri postupku provjere prava pristupa pojedinim operacijama usluga. Skup odredbi definira pružatelj usluge u kojima navodi uvjete korištenje registrirane usluge. Na osnovi podataka skup odredbi Sustav za nadzor pristupa odabire odgovarajuće mehanizme nadzora pristupa pojedinim uslugama. U skupu odredbi definiraju se operacije usluge, potreba za izvršavanjem provjere identiteta i prava pristupa korisnika te potreba za praćenjem korištenja usluge.

Sustav za nadzor pristupa sastoji se od Podsustava uspostave pristupa, Podsustava provedbe pristupa i Podsustava pozivatelj. Podsustav uspostave prima zahtjeve za registracijom korisnika i usluga te podatke koje unose administratori sustava na temelju kojih stvara i sprema upravljačke podatke. Podsustav provedbe pristupa na temelju raspoloživih upravljačkih podataka donosi odluku o prihvaćanju zahtjeva korištenja usluge. Podsustav pozivatelj smješta se lokalno na računalu pozivatelja i zadužen je za uspostavu, održavanje i ukidanje sjednice s Podsustavom uspostave pristupa.

Slika 4.6 prikazuje arhitekturu Sustava za nadzor pristupa. Na početku rada sustava administrator zadaje (1) ovlasti pristupa i upravljačke podatke. Primjenom Podsustava pozivatelj korisnici i pružatelji usluga registriraju (2, 2') vlastite



Slika 4.6: Arhitektura Sustava za nadzor pristupa

upravljačke podatke. Prije korištenja usluge korisnik se autentificira (3, 3') te dolazi do uspostave sjednice i slanja identifikatora sjednice Podsustavu pozivatelj. Podsustav pozivatelj umeće primljeni identifikator sjednice u svaki korisnikov zahtjev koji se šalje za vrijeme trajanja sjednice. Nakon uspostave sjednice, Podsustav uspostave pristupa prenosi (3'') sjedničke i upravljačke podatke Podsustavu provedbe pristupa na temelju kojih ovaj izvršava nadzor pristupa. Podsustav pozivatelj šalje (4) zahtjev za pristup usluzi. Podsustav provedbe pristupa provjerava valjanost primljenog zahtjeva na temelju spremljenih upravljački podataka. Ako je zahtjev u skladu sa spremljenim upravljačkim pravilima, prosljeđuje se usluzi (5). Usluga obrađuje korisnikov zahtjev i stvara odgovor koji šalje (6) Podsustavu provedbe pristupa. Podsustav provedbe pristupa bilježi korištenje usluge i prosljeđuje (7) odgovor Podsustavu pozivatelju.

5 Sustav za nadgledanje rada korisnika i usluga

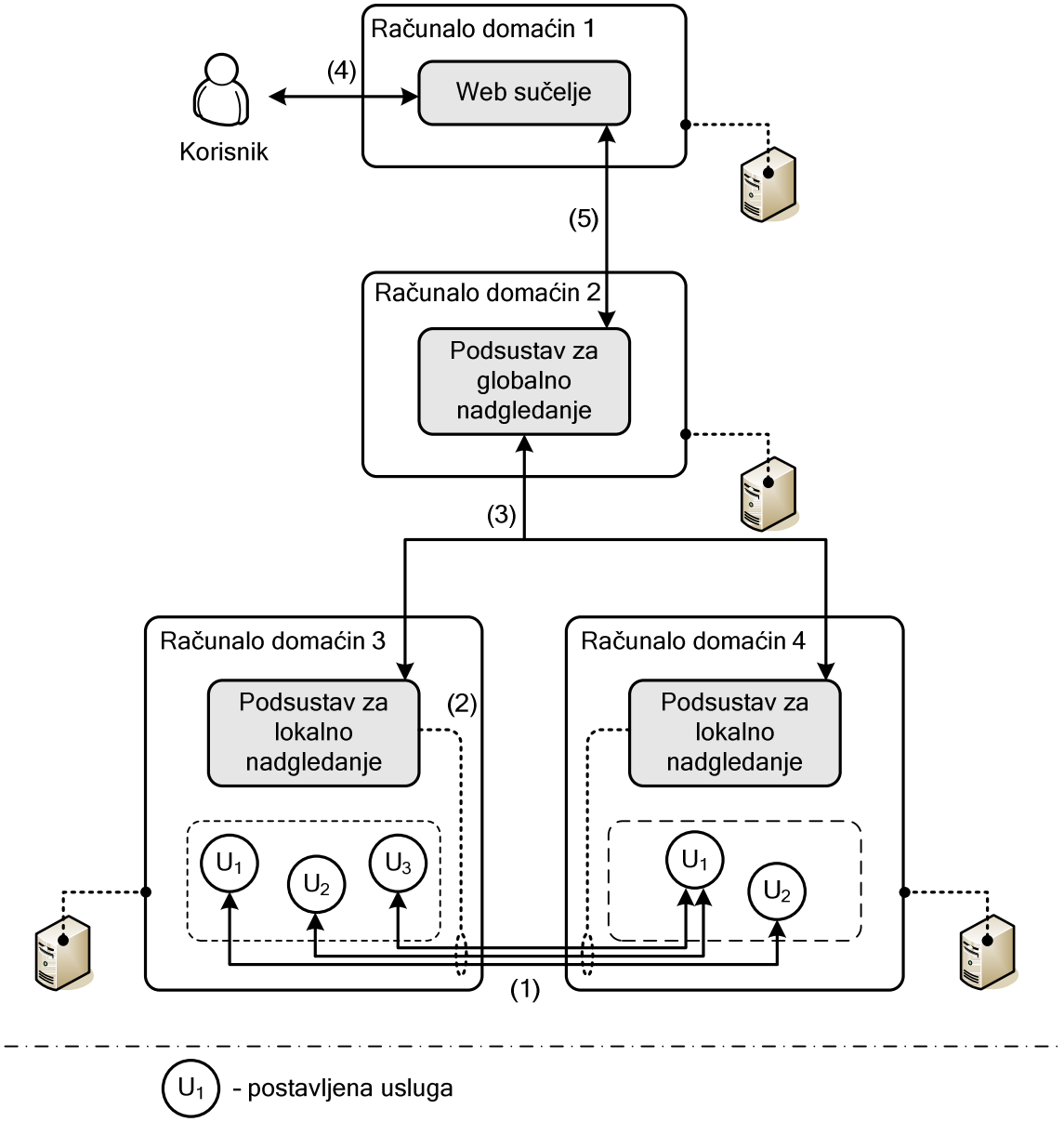
Sustavi zasnovani na uslugama sastoje se od skupa usluga koje tijekom rada međusobno izmjenjuju poruke s ciljem ostvarivanja željenih funkcionalnosti sustava. Zbog potrebe nadgledanja rada sustava i praćenja rada korisnika sustava, koristi se Sustav za nadgledanje rada korisnika i usluga. Sustav za nadgledanje rada korisnika i usluga ostvaruje presretanje poruka koje usluge razmjenjuju, analizu sadržaja poruka i izdvajanje sadržaja poruka s ciljem izgradnje zapisa o korištenju usluga. Stvoreni zapisi koriste se za ostvarivanje usluga naplaćivanja korištenja računalnih sredstava, sigurnosno nadgledanje rada korisnika, skladištenje podataka i rudarenje podataka. Arhitektura i programsko ostvarenje Sustava za nadgledanje rada korisnika i usluga opisani su u diplomskom radu [16].

U odjeljku 5.1 opisana je logička arhitektura Sustava za nadgledanje rada korisnika i usluga. Odjeljak 5.2 opisuje Podsustav za lokalno nadgledanje, dok odjeljak 5.3 opisuje Podsustav za globalno nadgledanje. U odjeljku 5.4 definiran je jezik za izgradnju obrazaca za nadgledanje, a u odjeljku 5.5 opisani su protokoli razmjene prikupljenih podataka.

5.1 Logička arhitektura

Budući da su usluge u sustavima zasnovanima na uslugama raspodijeljene Sustav za nadgledanje rada korisnika i usluga ima raspodijeljenu arhitekturu. Raspodijeljena arhitektura Sustava za nadgledanje rada korisnika i usluga sastoji se od Podsustava za lokalno nadgledanje, Podsustava za globalno nadgledanje i Web Sučelja. Podsustav za lokalno nadgledanje zadužen je za presretanje, analizu i izdvajanje sadržaja poruka koje usluge međusobno izmjenjuju. Izdvajanjem sadržaja poruka Podsustav za lokalno nadgledanje stvara zapise o korištenju usluga. Podsustav za globalno nadgledanje zadužen je za pribavljanje svih zapisa o korištenju usluga. Web sučelje omogućuje korisniku sustava uvid u zapise o korištenju pojedinih usluga i pruža mogućnost pretraživanje zapisa korištenjem različitih kriterija pretraživanja.

Slika 5.1 prikazuje logičku arhitekturu Sustava za nadgledanje rada korisnika i usluga. Tijekom rada sustava zasnovanih na uslugama usluge izmjenjuju poruke (1). Podsustavi za lokalno nadgledanje presreću poruke (2), analiziraju i izdvajaju sadržaj



Slika 5.1: Logička arhitektura Sustava za nadgledanje rada korisnika i usluga

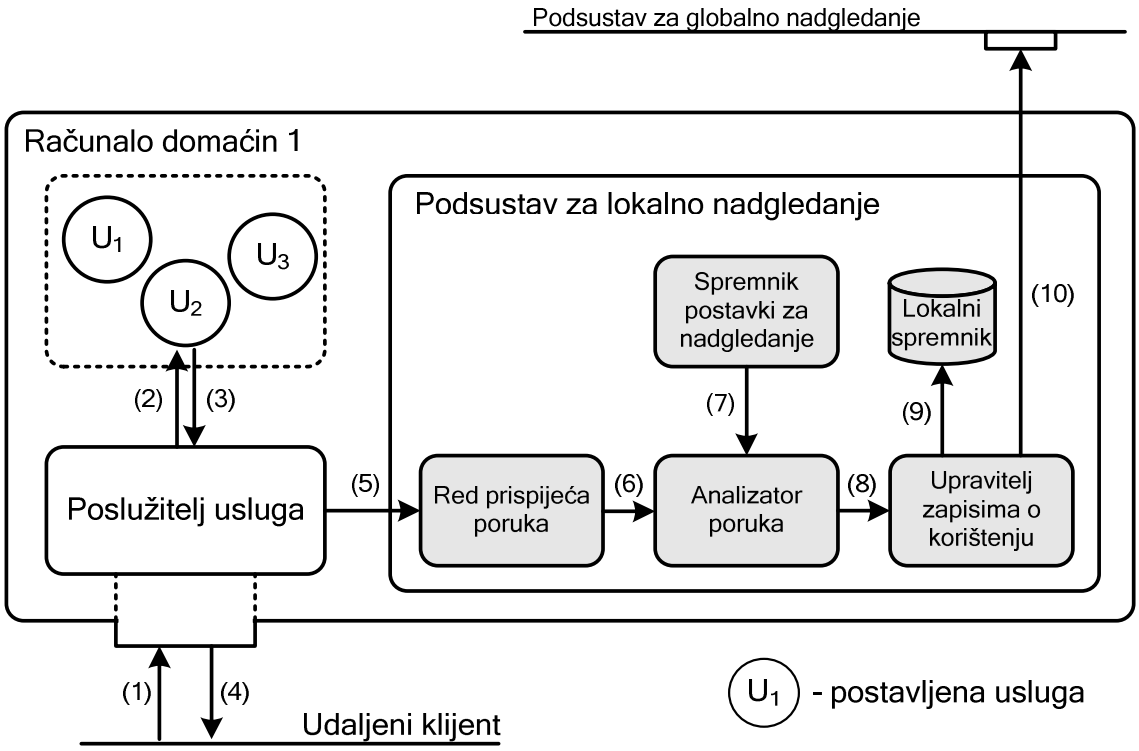
poruka. Izdvajanjem sadržaja poruka Podsustav za lokalno nadgledanje stvara zapise o korištenju usluga. Ovisno o korištenom protokolu za razmjenu podataka Podsustavi za lokalno nadgledanje i Podsustav za globalno nadgledanje razmjenjuju (3) zapise o korištenju ili adrese računala na kojima se nalaze zapisi o korištenju usluga. Upotrebom Web sučelja korisnik pristupa zapisima o korištenju usluga putem kojeg postavlja upite (4). Postavljeni upit Web sučelje prosljeđuje Podsustavu za globalno nadgledanje (5) koji prikuplja sve zapise o korištenju usluga i vraća ih natrag Web sučelju na prikaz korisniku.

5.2 Podsustav za lokalno nadgledanje

Podsustav za lokalno nadgledanje postavlja se na ona računala na kojima se nalaze usluge čiji se rad želi nadgledati. Nakon postavljanja usluga i Podsustava za lokalno nadgledanje potrebno je izvršiti prijavu usluge Podsustavu za lokalno nadgledanje. Prijava usluge ostvaruje se izgradnjom i prijavom odgovarajućeg obrasca za nadgledanje. Obrazac za nadgledanje određuje koji se dio poruka zahtjeva i odgovora usluge izdvaja. Nakon prijave usluge Podsustav za lokalno nadgledanje presreće poruke zahtjeva i odgovora prijavljene usluge, analizira i izdvaja sadržaj te stvara zapise o korištenju usluge. Radne postavke Sustava za nadgledanje rada korisnika i usluga određuju koji se protokol koristi za razmjenu prikupljenih podataka. Ovisno o protokolu razmjene prikupljenih podataka Podsustav za lokalno nadgledanje stvorene zapise o korištenju usluga sprema u lokalni spremnik računala domaćina ili ih sprema u lokalni spremnik Podsustava za globalno nadgledanje.

Podsustav za lokalno nadgledanje sastoji se od modula Rep poruka, modula Analizator poruka, modula Spremnik postavki za nadgledanje, modula Upravitelj zapisima o korištenju i modula Lokalni spremnik. Modul Red prispijeća poruka koristi se za spremanje svih poruka zahtjeva i odgovora usluga koje su proslijeđene Podsustavu za lokalno nadgledanje. Modul Analizator poruka analizira sadržaj poruka i ovisno o spremljenim postavkama za nadgledanje usluga izdvaja dijelove sadržaja poruka i zatim stvara zapise o korištenju usluga. Modul Spremnik postavki za nadgledanje služi za spremanje obrazaca za nadgledanje usluga. Modul Upravitelj zapisima o korištenju ovisno o radnim postavkama Sustava za nadgledanje rada korisnika i usluga zapise o korištenju usluga pohranjuje u modul Lokalni spremnik ili ih prosljeđuje Podsustavu za globalno nadgledanje. Modul Lokalni spremnik koristi se za spremanje zapisa o korištenju usluga.

Slika 5.2 prikazuje Podsustav za lokalno nadgledanje. Udaljeni klijent poziva (1) uslugu U_2. Pozivanje usluga na računalu domaćinu 1 omogućava Poslužitelj usluga koji izlaže usluge udaljenim klijentima. Poslužitelj usluga prosljeđuje poruku zahtjeva usluzi U_2, koja obrađuje zahtjev i stvara poruku odgovora koju zatim šalje (3) Poslužitelju usluga. Poslužitelj usluga prosljeđuje (4) poruku odgovora udaljenom



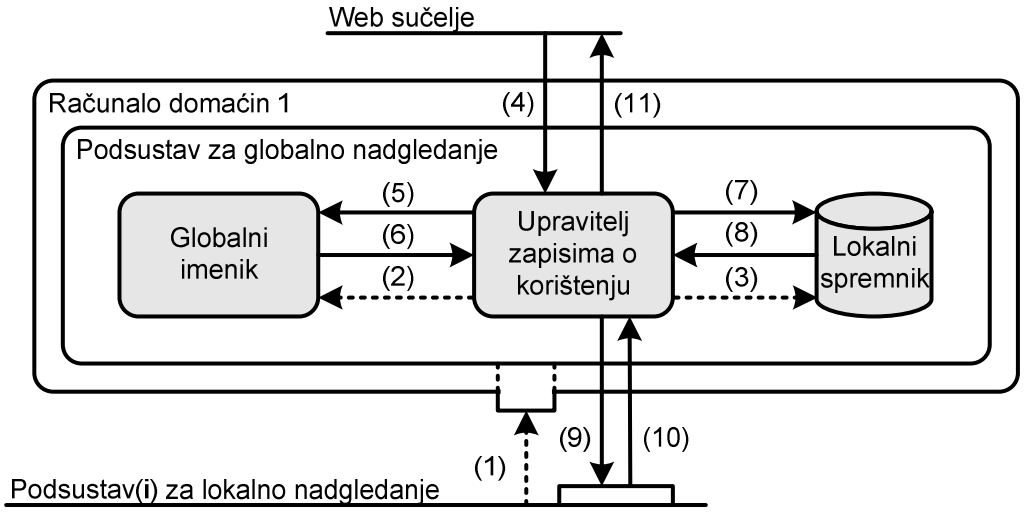
Slika 5.2: Podsustav za lokalno nadgledanje

klijentu. Zatim Poslužitelj usluga prosljeđuje (5) primljene poruke zahtjeva i odgovora Podsustavu za lokalno nadgledanje. Sve pristigle poruke spremaju se u modul Rep poruka prema redoslijedu prispijeća primjenom FIFO (engl. *First in first out*) podatkovne strukture. Modul analizator poruka preuzima poruke iz modula Red prispijeća poruka (6) koje zatim obrađuje. Ovisno o obrascima za nadgledanje (7), spremljenima u modulu Spremniku postavki za nadgledanje, modul Analizator poruka izdvaja sadržaj poruka i stvara zapise o korištenju usluga koje prosljeđuje (8) modulu Upravitelj zapisima o korištenju. Rad modula Upravitelj zapisima o korištenju sa zapisima o korištenju usluga ovisi o korištenom protokolu razmjene prikupljenih podataka. Ako se koristi protokol za istiskivanje modul Upravitelj zapisima o korištenju sprema (9) zapise o korištenju usluga u modul Lokalni spremnik i Podsustavu za globalno nadgledanje šalje (10) adresu računala domaćina 1. Ako se koristi protokol za potiskivanje modul Upravitelj zapisima o korištenju šalje zapise (10) o korištenju usluga Podsustavu za globalno nadgledanje.

5.3 Podsustav za globalno nadgledanje

Podsustav za globalno nadgledanje omogućuje prikupljanje zapisa o korištenju usluga na jedno mjesto olakšavajući time njihovu obradu i analizu.

Podsustav za globalno nadgledanje sastoji se od modula Globalni imenik, modula Upravitelj zapisima o korištenju i modula Lokalni spremnik. Modul Globalni imenik sadrži informacije o lokacijama na kojima se nalaze zapisi o korištenju usluga. Modul Upravitelj zapisima o korištenju ovisno o primljenim podacima od strane Podsustava za lokalno nadgledanje sprema vrijednosti u modul Globalni imenik i modul Lokalni spremnik. Podsustav za globalno nadgledanje također obrađuje korisnikove upite te korištenjem informacija iz modula Globalni imenik dohvaća potrebne zapise o korištenju usluga i vraća rezultate upita. U slučaju korištenja protokola za potiskivanje modul Lokalni spremnik koristi se za spremanje zapisa o korištenju usluga, a u slučaju korištenja protokola za istiskivanje ne koristi se.



Slika 5.3: Podsustav za globalno nadgledanje

Slika 5.3 prikazuje Podsustav za globalno nadgledanje. Tijekom rada Podsustavi za lokalno nadgledanje stvaraju zapise o korištenju usluga. Stvoreni zapisi o korištenju usluga spremaju se lokalno ili na računalu domaćinu Podsustava za lokalno nadgledanje. Ako se zapisi spremaju lokalno onda Podsustavi za lokalno nadgledanje šalju (1) adresu računala domaćina koju modul Upravitelj zapisima o korištenju sprema (2) u modul Globalni imenik. Ako se šalju zapisi o korištenju usluga modul Upravitelj zapisima o korištenju prijavljuje (2) primljene zapise o korištenju usluge modulu Globalni imenik i zatim ih sprema (3) u modul Lokalni spremnik. Korisnik sustava upotrebom Web sučelja postavlja upit koji se sastoji od ključnih riječi koje Podsustav za globalno nadgledanje koristi za indeksiranje zapisa o korištenju usluga. Korisnikov upit Web sučelje prosljeđuje (4) Podsustavu za globalno nadgledanje. Modul Upravitelj zapisima o korištenju dohvaća (5,6) informacije o

lokacijama na kojima se nalaze zapisi o korištenju usluga. Ako su zapisi o korištenju usluga spremljeni u modulu Lokalni spremnik, modul Upravitelj zapisima o korištenju pribavlja odgovarajuće zapise o korištenju usluga (7,8) iz modula Lokalni Spremnik. Ako su zapisi o korištenju usluga spremljeni na računalima domaćinima Podsustava za lokalno nadgledanje onda modul Upravitelj zapisima o korištenju šalje upit svim računalima (9) čija se adresa nalazi u modulu Globalni imenik. Svaki od Podsustava za lokalno nadgledanje obrađuje korisnikov upit i dostavlja (10) odgovarajuće zapise o korištenju usluge. Pribavljeni zapisi o korištenju usluge prosljeđuju (11) se Web sučelju.

5.4 Jezik za izgradnju obrazaca za nadgledanje

Podsustavi za lokalno nadgledanje analiziraju i izdvajaju sadržaj poruka na temelju obrazaca za nadgledanje. Obrasce za nadgledanje grade vlasnici usluga, za one usluge čiji se rad želi pratiti, upotrebom jezika za izgradnju obrazaca za nadgledanje. Obrazac za nadgledanje je XML datoteka koja sadrži informacije o tome koje je elemente SOAP poruka zahtjeva i odgovora potrebno izdvajati. Slika 5.4 prikazuje općenitu građu obrasca za nadgledanje.

```
<ServiceAccountFormat>
   <serviceId>
      <functionId>
         <request>
            <header>
               <trackPath> XPath </trackPath>
            </header>
            <body>
               <trackPath> XPath </trackPath>
            </body>
         </request>
         <response>
            <header>
               <trackPath> XPath </trackPath>
            </header>
            <body>
               <trackPath> XPath </trackPath>
            </body>
         </response>
      </functionId>
   </serviceId>
</ServiceAccountFormat>
```

Slika 5.4: Općenita građa obrasca za nadgledanje

Korijenski element obrasca za nadgledanje je *<ServiceAccountFormat>*. Unutar korijenskog elementa ugnježđuje se element *<serviceId>* pri čemu *serviceId*

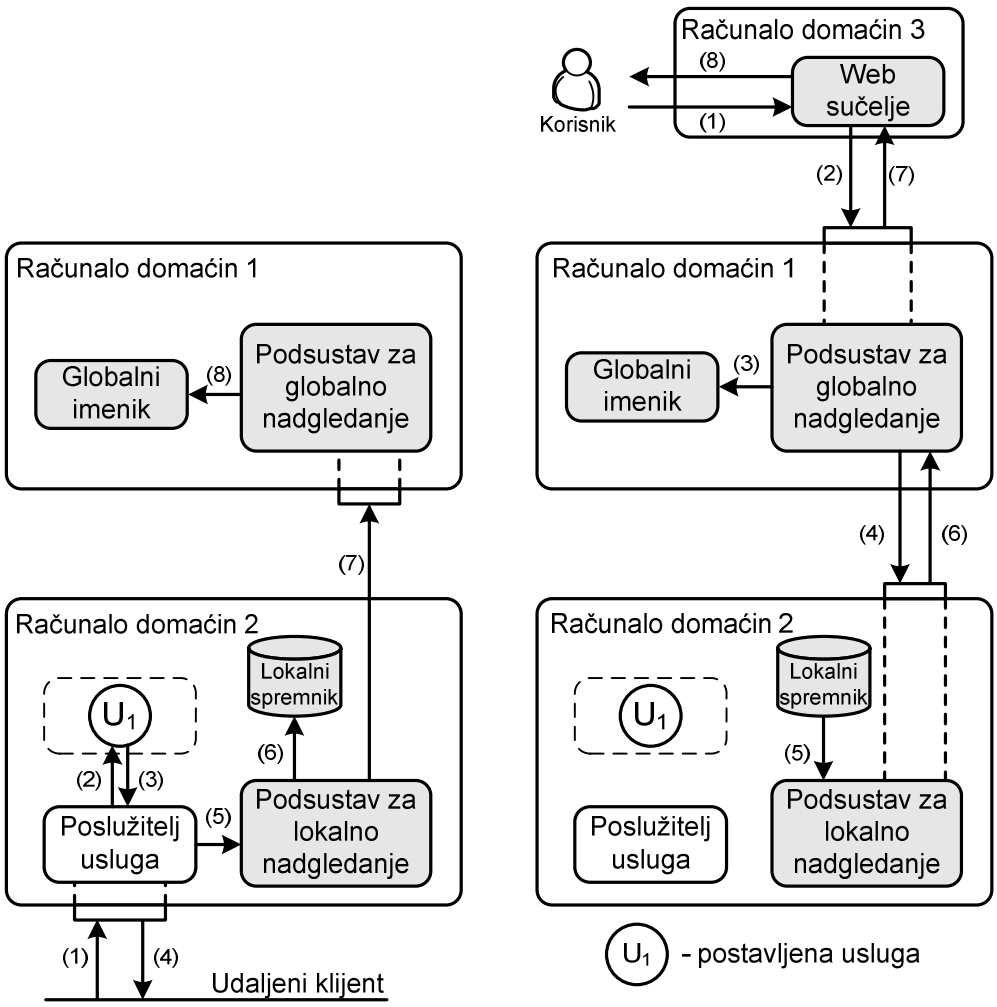
predstavlja naziv usluge za koju se gradi obrazac za nadgledanje. Unutar *<serviceId>* elementa ugnježđuje se element *<functionId>* pri čemu *functionId* predstavlja naziv metode usluge za koju se gradi obrazac za nadgledanje. Unutar *<functionId>* elementa moguće je ugnijezditi dva elementa *<request>* i *<response>*. Element *<request>* koristi se pri izdvajanju sadržaja SOAP poruka zahtjeva, a element *<response>* pri izdvajanju sadržaja SOAP poruka odgovora. Unutar elemenata *<request>* i *<response>* moguće je ugnijezditi elemente *<header>* i *<body>*. Element *<header>* koristi se za izdvajanje sadržaja zaglavlja SOAP poruka, a element *<body>* za izdvajanje sadržaja tijela SOAP poruka. Elementi *<header>* i *<body>* sadrže *XPath* izraz koji definira koji se elementi zaglavlja i tijela SOAP poruka moraju izdvojiti. Tijekom izrade obrasca za nadgledanje nije potrebno navesti sve *XPath* izraze.

5.5 Protokoli razmjene prikupljenih podataka

Sustav za nadgledanje rada korisnika i usluga podržava dva protokola razmjene prikupljenih podataka koje Podsustavi za lokalno nadgledanje i Podsustav za globalno nadgledanje koriste za razmjenu zapisa o korištenju usluga. Podržani su protokol za istiskivanje i protokol za potiskivanje. Osnovna razlika između dva protokola jest lokacija na koju se spremaju zapisi o korištenju usluga.

Protokol za istiskivanje zasniva se na istiskivanju zapisa o korištenju usluga. Tijekom rada Podsustav za lokalno nadgledanje stvara zapise o korištenju usluga koje pohranjuje na računalu domaćinu. Na zahtjev korisnika Podsustav za globalno nadgledanje dohvaća zapise o korištenju usluga. Slika 5.5 prikazuje postupke spremanja i dohvata zapisa o korištenju usluga primjenom protokola za istiskivanje.

Slika 5.5 a) prikazuje primjenu protokola za istiskivanje u svrhu spremanja zapisa o korištenju usluga tijekom rada Sustava za nadgledanje rada korisnika i usluga. Udaljeni klijent poziva uslugu U_1 na računalu domaćinu 2 (1,2,3,4). Po završetku poziva usluge U_1 Poslužitelj usluga prosljeđuje (5) poruke zahtjeva i odgovora Podsustavu za lokalno nadgledanje. Podsustav za lokalno nadgledanje stvara zapise o korištenju usluge koje sprema (6) u modul Lokalni spremnik na računalu domaćinu 2. Zatim šalje (7) adresu računala domaćina 2 Podsustavu za globalno nadgledanje koji primljenu adresu sprema (8) u modul Globalni imenik.



a) spremanje zapisa b) dohvat zapisa

Slika 5.5: Spremanje i dohvat zapisa uz korištenje protokola za istiskivanje

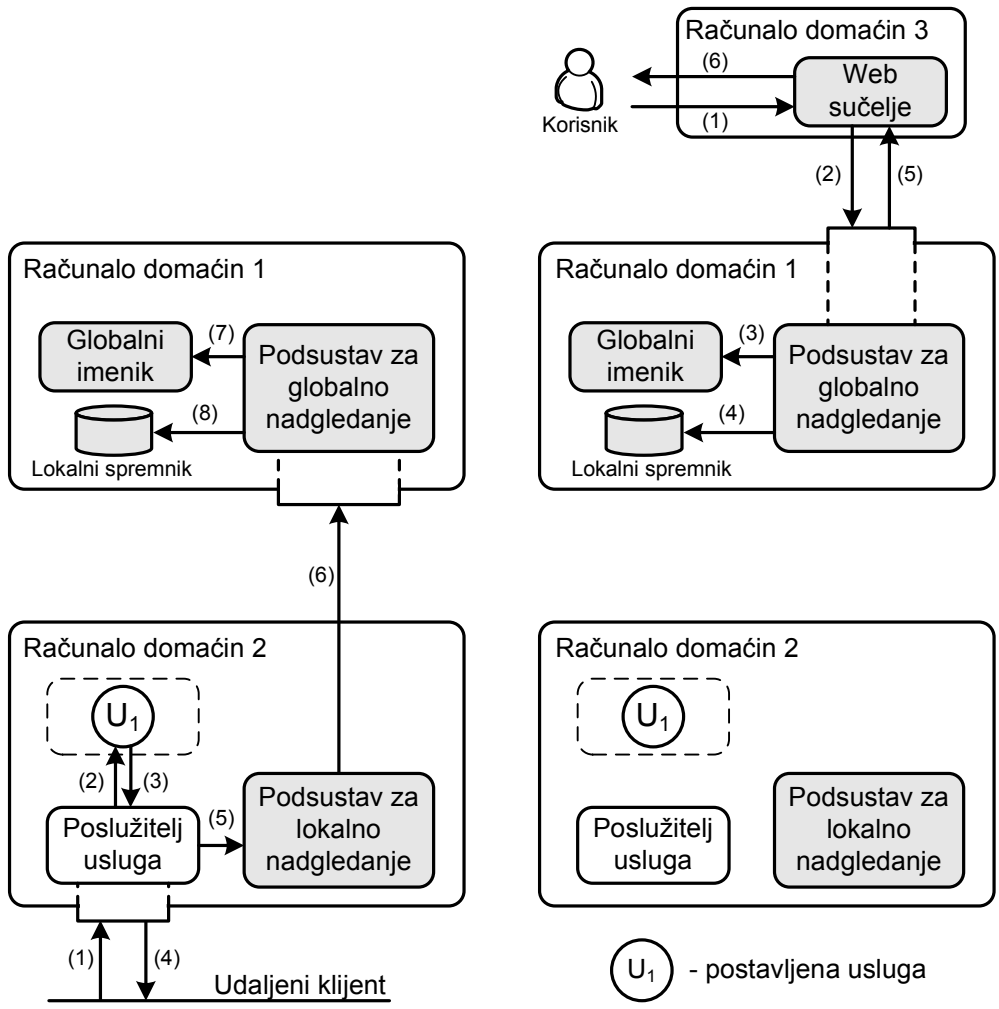
Slika 5.5 b) prikazuje dohvat zapisa o korištenju usluga na zahtjev korisnika sustava uz primjenu protokola za istiskivanje. Korisnik sustava putem Web sučelja postavlja (1) upit koji se prosljeđuje (2) Podsustavu za globalno nadgledanje. Podsustav za globalno nadgledanja na temelju informacija iz modula Globalni imenik (3) dohvaća adresu računala na kojem se nalaze zapisi o korištenju usluga. Podsustav za globalno nadgledanje prosljeđuje (4) korisnikov upit Podsustavu za lokalno nadgledanje na računalo domaćin 2 čiju je adrese dohvatio iz modula Globalni imenik. Podsustav za lokalno nadgledanje na temelju korisnikova upita dohvaća (5) zapise iz modula Lokalni spremnik i šalje (6) ih Podsustavu za globalno nadgledanje. Podsustav za globalno nadgledanje prosljeđuje (7) primljene zapise o korištenju usluga Web sučelju koje ih zatim prikazuje (8) korisniku.

Protokol za istiskivanje stvara mali mrežni promet budući da se zapisi o korištenju usluga spremaju na računalu domaćinu Podsustava za lokalno nadgledanje, a putem računalne mreže prenosi se samo adresa računala domaćina Podsustava za lokalno nadgledanje. Međutim, nedostatak protokola za istiskivanje jest povećano zauzeće računalnih sredstava na računalima domaćinima Podsustava za lokalno nadgledanje što uzrokuje manju učinkovitost ostalih usluga smještenih na tom istom računalu. Nadalje, tijekom dohvata zapisa o korištenju usluga povećan je mrežni promet budući da dolazi do prijenosa zapisa o korištenju usluga.

Protokol za potiskivanje zasniva se na spremanju zapisa o korištenju usluga u središnji spremnik Podsustava za globalno nadgledanje. Tijekom rada Podsustav za lokalno nadgledanje stvara zapise o korištenju usluga koje nakon stvaranja prosljeđuje Podsustavu za globalno nadgledanje na spremanje. Slika 5.6 prikazuje postupke spremanje i dohvata zapisa o korištenju usluga primjenom protokola za potiskivanje.

Slika 5.6 a) prikazuje spremanje zapisa o korištenju usluga tijekom rada Sustava za nadgledanje rada korisnika i usluga uz primjenu protokola za potiskivanje. Udaljeni klijent poziva usluga U_1 na računalu domaćinu 2 (1,2,3,4). Po završetku poziva usluge U_1 Poslužitelj usluga prosljeđuje (5) poruke zahtjeva i odgovora Podsustavu za lokalno nadgledanje. Podsustav za lokalno nadgledanje stvara zapise o korištenju usluge koje prosljeđuje (6) Podsustavu za globalno nadgledanje. Podsustav za globalno nadgledanje ažurira (7) vrijednosti u modulu Globalni imenik i sprema (8) zapise o korištenju usluga u modul Lokalni spremnik.

Slika 5.6 b) prikazuje dohvat zapisa o korištenju usluga na zahtjev korisnika sustava uz primjenu protokola za potiskivanje. Korisnik sustava putem Web sučelja postavlja (1) upit koji se prosljeđuje (2) Podsustavu za globalno nadgledanje. Na temelju informacija iz modula Globalni imenik (3) Podsustav za globalno nadgledanje saznaje da su zapisi o korištenu usluga spremljeni u modulu Lokalni spremnik Podsustava za globalno nadgledanje. Na temelju korisničkog upita dohvaća (4) odgovarajuće zapise o korištenju usluga iz modula Lokalni spremnik koje zatim šalje (5) Web sučelju. Web sučelje prikazuje (6) korisniku zapise o korištenju usluga koji su rezultat korisničkog upita.



a) spremanje zapisa b) dohvat zapisa

Slika 5.6: Spremanje i dohvat zapisa uz korištenje protokola za potiskivanje

Protokol za potiskivanje omogućava učinkovit dohvat rezultata izračuna korisnikova upita što je posljedica činjenice da su svi korisnički zapisi pohranjeni na računalu domaćinu Podsustava za globalno nadgledanje. Međutim, nedostatak protokola za potiskivanje su zahtjev za većim spremnikom računala domaćina Podsustava za globalno nadgledanje i stvaranje dodatnog mrežnog promet koji opterećuje sustav tijekom rada sustava budući da se prijenose cjelokupni zapisi o korištenju usluga.

6 Sustav za ispitivanje radnih značajki usluga

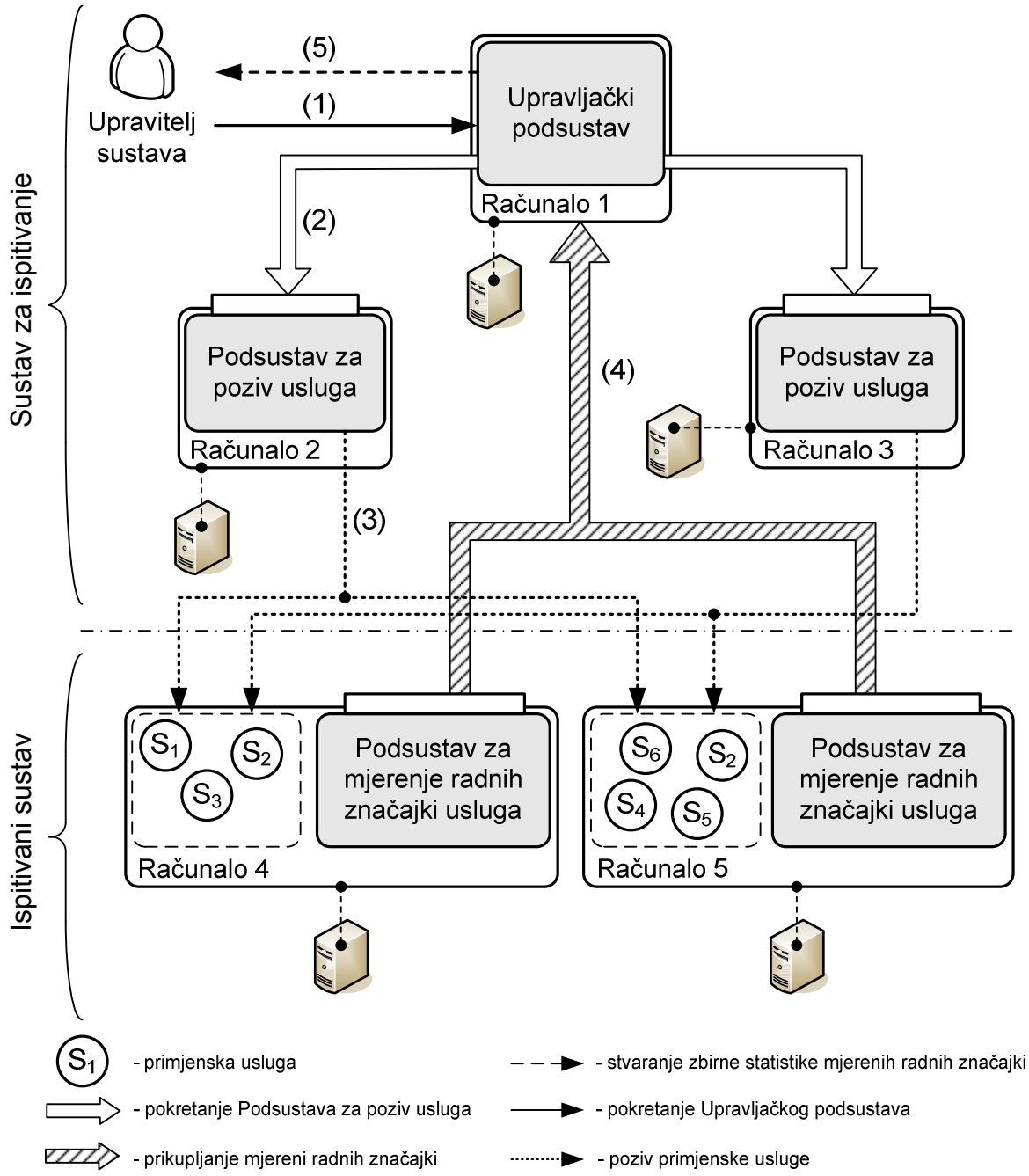
Usluge zauzimaju različita računalna sredstva računala domaćina od trenutka prispijeća poruke zahtjeva, tijekom obrade pristiglog zahtjeva i prosljeđivanja poruke odgovora. Radnu memoriju usluge zauzimaju za spremanje ulaznih parametara, za spremanje programskog koda koji se treba izvršiti u sklopu poziva usluge i za spremanje lokalnih varijabli korištenih za vrijeme obrade zahtjeva. Diskovni prostor se zauzima za spremanje vrijednosti koje je potrebno sačuvati između dva poziva primjenske usluge. Izvođenje programskog koda pozvanih metoda primjenskih usluga zauzima procesorsko vrijeme. Kako bi se omogućilo mjerenje i vrednovanje radnih značajki sustava zasnovanih na uslugama razvijen je Sustav za ispitivanje radnih značajki usluga. Sustav za ispitivanje radnih značajki usluga sastoji se od skupa usluga koje zajednički mjere i prikupljaju informacije o iskorištenju računalnih sredstava računala domaćina.

Sustav za ispitivanje radnih značajki usluga ostvaren je u sklopu projekta "*Okrilje posrednika javnog informacijskog sustava*" koji je dio poliprojekta CroGrid potpomognutog od strane Ministarstva znanosti, obrazovanja i športa Republike Hrvatske. Poliprojekt CroGrid pokrenut je na Fakultetu elektrotehnike i računarstva u suradnji s tvrtkom Ericsson Nikola Tesla d.d. u sklopu kojeg je oblikovan i ostvaren PIE raspodijeljeni sustav zasnovan na uslugama.

U odjeljku 6.1 opisana je logička arhitektura Sustava za ispitivanje radnih značajki usluga, a u odjeljku 6.2 programsko ostvarenje Sustava za ispitivanje radnih značajki.

6.1 Logička arhitektura sustava

Sustavi zasnovani na uslugama sastoje se od skupa slabo povezanih usluga u raspodijeljenoj računalnoj okolini. Usluge međusobno razmjenjuju poruke s ciljem ostvarivanja funkcionalnosti raspodijeljenog sustava. Budući da su sustavi zasnovani na uslugama raspodijeljeni razvijena je raspodijeljena arhitektura sustava za ispitivanje radnih značajki usluga. Ostvarena raspodijeljena arhitektura sastoji se od sljedećih podsustava: Upravljačkog podsustava, Podsustava za poziv usluga i Podsustava za mjerenje radnih značajki usluga.



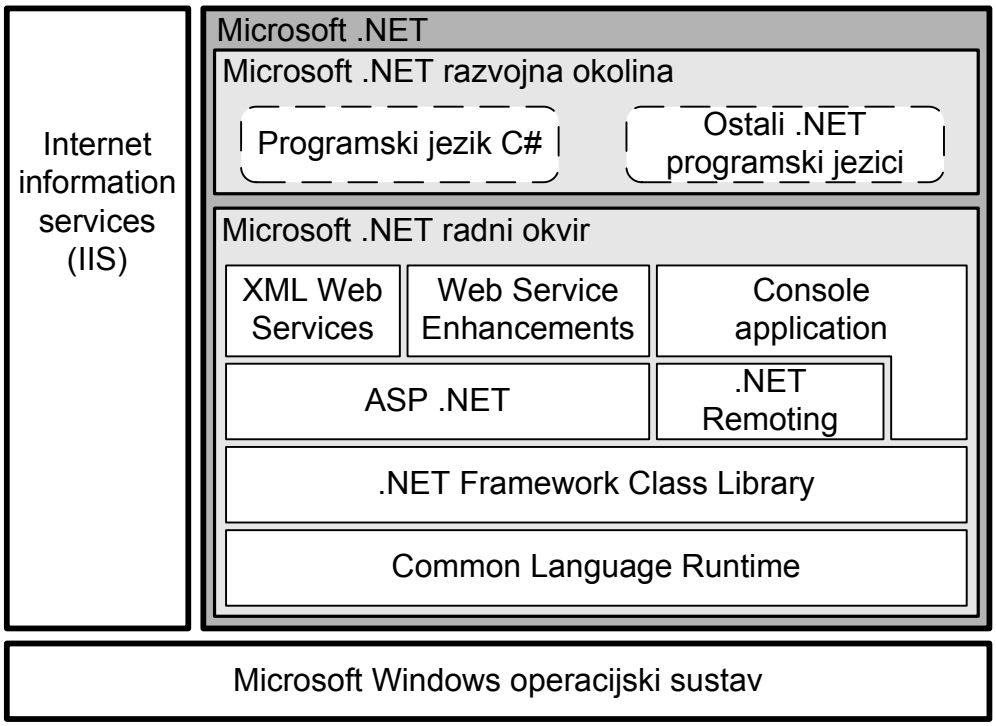
Slika 6.1: Primjer postavljanja Sustava za ispitivanje radnih značajki usluga

Upravljački podsustav učitava radne postavke, pokreće Podsustave za mjerenje radnih značajki i Podsustave za poziv usluga, zaustavlja Podsustave za mjerenje radnih značajki, prikuplja i stvara zbirnu statistiku mjerenih radnih značajki. Podsustav za poziv usluga poziva ispitne usluge u zadanim vremenskim intervalima kroz zadani vremenski period. Podsustav za mjerenje radnih značajki sustava prikuplja informacije o iskorištenju računalnih sredstava računala domaćina.

Na slici 6.1 prikazan je primjer postavljanja Sustava za ispitivanje radnih značajki usluga. Prikazani raspodijeljeni sustav sastoji se od 5 računala. Na računalu 1 postavljen je Upravljački podsustav, dok je na računalima 2 i 3 postavljen Podsustav za poziv usluga. Na računalima 4 i 5 postavljeni su Podsustav za mjerenje radnih značajki usluga i primjenske usluge (S_1 – S_6). Upravitelj sustava određuje radne postavke Sustava za ispitivanje radnih značajki i zatim pokreće (1) Upravljački podsustav. Upravljački podsustav učita radne postavke, prosljeđuje ih Podsustavima za poziv usluga i zatim pokreće pozivanje primjenskih usluga (2). Podsustavi za poziv usluga (3) ostvaruju pozive prema primjenskim uslugama na računalima 4 i 5. Tijekom poziva primjenskih usluga, Podsustavi za mjerenje radnih značajki usluga očitavaju statistiku zauzeća računalnih sredstava na računalima domaćinima i očitane rezultate spremaju lokalno. Po završetku ispitivanja, Upravljački podsustav (4) prikuplja podatke od Podsustava za mjerenje radnih značajki usluga, sa računala 4 i 5, i stvara zbirnu statistiku mjerenih radnih značajki ispitivanog sustava koju prosljeđuje (5) Upravitelju sustava.

6.2 Programsko ostvarenje

Sustav za ispitivanje radnih značajki usluga ostvaren je primjenom Microsoft *.NET* razvojnog okruženja [17]. Logička arhitektura Microsoft *.NET* razvojnog okruženja prikazana je na slici 6.2. Microsoft *.NET* razvojno okruženje omogućuje oblikovanje, razvoj i izvršavanje primjenskih programa, a sastoji se od Microsoft *.NET* razvojne okoline i Microsoft *.NET* radnog okvira (engl. *framework*). Microsoft *.NET* razvojna okolina omogućuje oblikovanje i razvijanje primjenskih programa korištenjem različitih *.NET* programskih jezika. Za svaki od *.NET* programskih jezika dostupan je odgovarajući jezični procesor. Ciljni jezik *.NET* programskih prevodilaca je IL (engl. *Intermediate Language*) programski jezik. IL programski jezik temelji se na strojno nezavisnom kodu za čije je izvršavanje potrebno imati izvršnu okolinu zasnovanu na CLI (engl. *Common Language Infrastructure*) standardu. CLI standard određuje značajke infrastrukture koja izvršava programe napisane u IL programskom jeziku. Microsoft infrastruktura za izvršavanje programa pisanih u IL programskom jeziku naziva se Common Language Runtime (CLR) i ostvarena je u sklopu Microsoft *.NET* radnog okvira.



Slika 6.2: Osnovni elementi i korišteni programski alati razvojnog okruženja *.NET*

Microsoft *.NET* radni okvir sastoji se od više podsustava od kojih dva predstavljaju osnovu za razvijanje i izvršavanje primjenskih programa. To su podsustavi *.NET* Framework Class Library (FCL) i Common Language Runtime (CLR). Podsustav FCL sastoji se od skupa razreda, sučelja i vrijednosnih tipova (engl. *value types*) koji olakšavaju razvojni proces i pružaju pristup osnovnim funkcionalnostima računalnog sustava. Podsustav CLR zadužen je za izvođenje programa napisanih u IL programskom jeziku. Osim što izvodi programe pisane u IL programskom jeziku CLR je zadužen za obavljanje niz drugih operacija kao što su upravljanje memorijom, dretvama i greškama koje osiguravaju sigurno izvođenja primjenskog programa.

Povrh podsustava *.NET* Framework Class Library i Common Language Runtime nalaze se ostali podsustavi Microsoft *.NET* radnog okvira. Podsustav *ASP .NET* [18] omogućuje izgradnju i izvođenje dinamičkih Web primjenskih programa. Primjenski programi izgrađeni korištenjem *ASP .NET* podsustava dostupni su na korištenje u globalnoj mreži Internet putem IIS (engl. *Internet Information Services*) sustava [19]. Sustav Internet Information Services omogućuje postavljanje, izvođenje i upravljanje poslužiteljima usluga u globalnoj mreži Internet. Osnovne funkcionalnosti *ASP .NET* podsustava proširuje podsustav XML Web Services koji omogućuje

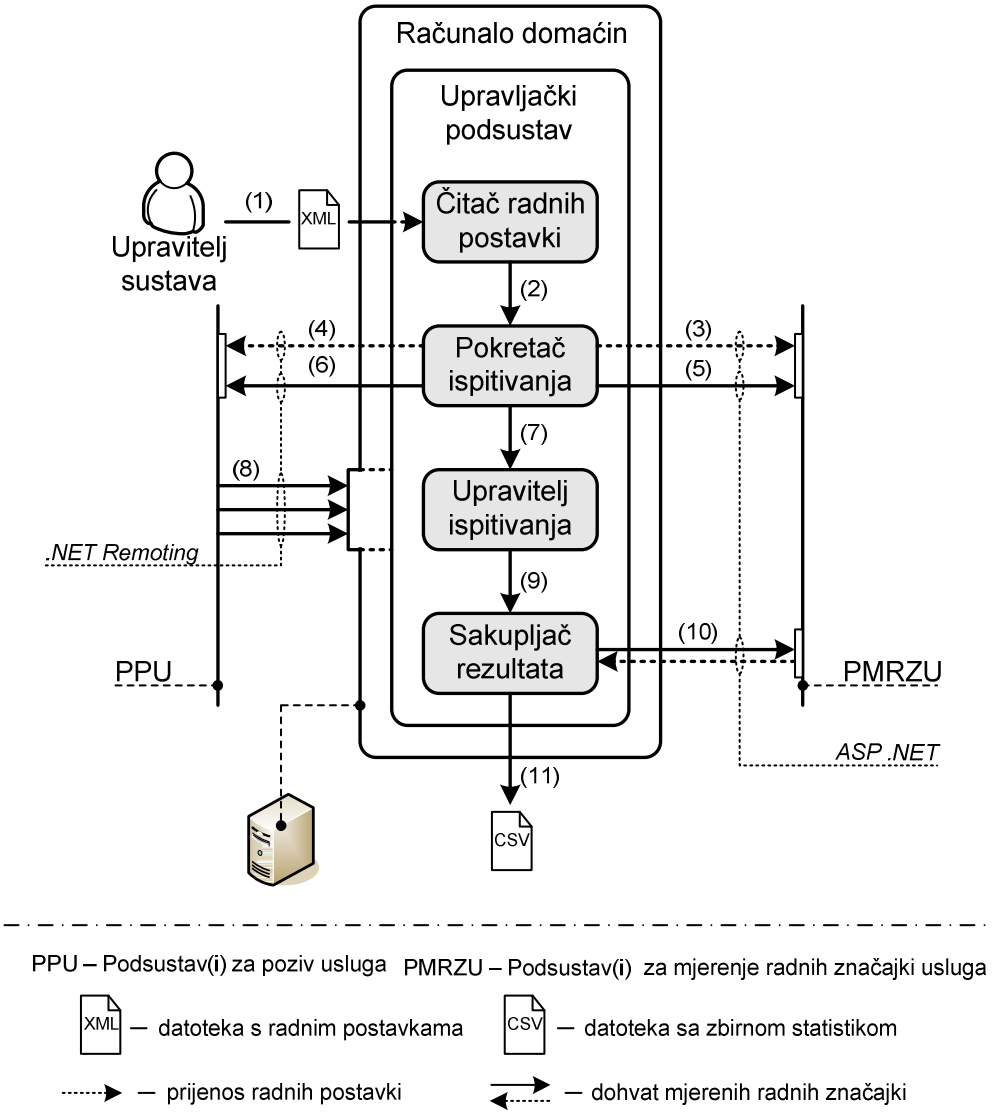
razvoj, postavljanje, izvođenje i upravljanje uslugama zasnovanim na Web Services tehnologiji. Napredne funkcionalnosti Web Services tehnologije ostvarene su u programskoj knjižici Web Services Enhancements [20]. Podsustav *.NET Remoting* omogućuje izradu raspodijeljenih programa u kojima se klijent odvaja od poslužitelja i od mehanizma komunikacije. Razvijatelju programske potpore pružena je mogućnost određivanja komunikacijskog protokola i načina na koji se informacije i podaci prenose. Podsustav Console Application omogućuje izgradnju primjenskih programa s komandno-linijskim sučeljem zasnovanih na podsustavima koji se, u hijerarhijskom smislu, nalaze ispod njega.

6.2.1 Upravljački podsustav

Upravljački podsustav je središnji podsustav Sustava za ispitivanje radnih značajki usluga koji nadgleda i upravlja radom ostalih podsustava Sustava za ispitivanje radnih značajki usluga. Zadaci Upravljačkog podsustava su učitavanje postavki za upravljanje radom sustava, pokretanje Podsustava za poziv usluga, pravovremeno pokretanje i zaustavljanje Podsustava za mjerenje radnih značajki usluga i prikupljanje rezultata s mjerenim radnim značajkama. Upravljački podsustav sastoji se od modula Čitač radnih postavki, modula Pokretač ispitivanja, modula Upravitelj ispitivanja i modula Sakupljač rezultata.

Modul Čitač radnih postavki učitava radne postavke koje je definirao Upravitelj sustava pri pokretanju Upravljačkog podsustava. Modul Pokretač ispitivanja prenosi radne postavke i pokreće Podsustave za mjerenje radnih značajki usluga i Podsustave za poziv usluga. Modul Upravitelj ispitivanja čeka kraj ispitivanja i trenutak kad su svi Podsustavi za poziv usluga javili istek vremenskog trajanja ispitivanja. Modul Sakupljač rezultata prikuplja mjerene radne značajke od svih Podsustava za mjerenje radnih značajki usluga, gradi zbirnu statistiku koju predaje Upravitelju sustava.

Slika 6.3 prikazuje programsku arhitektura Upravljačkog podsustava. Upravitelj sustava prije pokretanja Upravljačkog podsustava stvara XML datoteku s postavkama za upravljanje radom sustava. Nakon stvaranja datoteke s radnim postavkama Upravitelj sustava (1) pokreće Upravljački podsustav. Nakon pokretanja, modul Čitač radnih postavki učitava i parsira predanu XML datoteku i učitane radne postavke predaje (2) modulu Pokretač ispitivanja. Modul Pokretač



Slika 6.3: Programska arhitektura Upravljačkog podsustava

ispitivanja prenosi radne postavke Podsustavima za mjerenje radnih značajki usluga (3) i Podsustavima za poziv usluga (4). Modul Pokretač ispitivanja pokreće (5) Podsustave za mjerenje radnih značajki usluga i (6) Podsustave za poziv usluga. Komunikacija s Podsustavima za poziv usluga ostvarena je primjenom *.NET* Remoting tehnologije, dok je komunikacija s Podsustavima za mjerenje radnih značajki usluga ostvarena primjenom SOAP protokola iz skupa Web Services tehnologija. Modul Upravitelj ispitivanja (7) čeka kraj ispitivanja. Kraj ispitivanja nastupa kad se svaki od pokrenutih Podsustava za poziv usluga javi (8) s porukom o isteku vremenskog perioda ispitivanja. Podsustavi za poziv usluga pozivaju metodu *TestEnd* sučelja Upravljačkog podsustava. Sučelje Upravljačkog podsustava prikazano je u tablici 6.1. Nakon trenutka kraja ispitivanja (9) modul Sakupljač rezultata prikuplja (10) sve raspoložive rezultate s mjerenim radnim značajkama. Po

završetku prikupljanja rezultata modul Sakupljač rezultata stvara (11) izlaznu datoteku s prikupljenim rezultatima mjerenih radnih značajki.

Ime	Ulazni parametar	Izlazni parametar
TestEnd	Adresa računala domaćina Podsustava za poziv usluga	-

Tablica 6.1: Sučelje Upravljačkog podsustava

6.2.2 Postavke za upravljanje radom sustava

Postavke za upravljanje radom sustava spremaju se u obliku XML datoteke koja se predaje Upravljačkom podsustavu pri pokretanju. Struktura XML datoteke s postavkama za upravljanje radom sustava prikazana je na slici 6.4. Vršni element

```
<config>
	<tcaddress>adresa</tcaddress>
	<testrepetitions>N</testrepetitions>

	<pemconfigurations>
		<pemconfiguration address="">
			...
		</pemconfiguration>
		...
	</pemconfigurations>

	<loadgenerators opmode="">
		<loadgenerator address="">
			...
		</loadgenerator>
		...
	</loadgenerators>
</config>
```

Slika 6.4: Struktura XML datoteke s radnim postavkama

XML dokumenta je *<config>* element. Unutar <config> elementa ugnježđuju se svi ostali elementi XML datoteke s radnim postavkama. Element *<tcaddress>* sadrži adresu računala na kojem se nalazi Upravljački podsustav. Element *<testrepetitions>* definira broj ispitnih sljedova koje će Upravljački podsustav izvesti. Ispitni slijed predstavlja cjelokupan ciklus pokretanja ispitivanja na Podsustavima za poziv usluga koji se koriste u sklopu ispitnog sustava, čekanja na istek vremenskog perioda ispitivanja, prikupljanja rezultata mjerenih radnih značajki i njihovog zapisivanja u izlaznu datoteku. Zatim slijede elementi *<pemconfigurations>* i *<loadgenerators>* od kojih svaki ima konačan broj ugniježđenih elemenata. Unutar elementa *<pemconfigurations>* može se nalaziti konačan broj ugniježđenih elemenata

<pemconfiguration> koji definiraju radne postavke Podsustava za mjerenje radnih značajki usluga. Unutar elementa *<loadgenerators>* može se nalaziti konačan broj ugniježđenih elemenata *<loadgenerator>* koji definiraju radne postavke Podsustava za poziv usluga. Elementu *<loadgenerators>* potrebno je odrediti vrijednost atributa *opmode* koja određuje koja će se tehnologija koristiti za pozivanje ispitnih usluga koje su u sklopu ispitnog sustava. Moguće vrijednosti atributa *opmode* su *ws* i *overlay*.

Sadržajem elementa *<pemconfiguration>* i njegovih ugniježđenih elemenata određujemo postavke za upravljanje radom Podsustava za mjerenje radnih značajki usluga. Slika 6.5 prikazuje primjer postavki elementa *<pemconfiguration>.*

```
<pemconfiguration address="adresa">
       <cpu>true ili false</cpu>
       <networkadapter>true ili false</networkadapter>
       <filemonitor name="relativna staza">true ili false</filemonitor>
       <sampleperiod>T</sampleperiod>
</pemconfiguration>
```

Slika 6.5: Primjer postavki elementa *<pemconfiguration>*

Atribut *address*, elementa *<pemconfiguration>*, sadrži adresu Podsustava za mjerenje radnih značajki kojem pridružujemo definirane postavke za upravljanje radom sustava. Elementu postavki *<pemconfiguration>* potrebno je odrediti vrijednost atributa *address* koja predstavlja adresu računala na kojem se nalazi Podsustav za mjerenje radnih značajki usluga koji se želi koristiti. Unutar elementa *<pemconfiguration>* nalaze se elementi *<cpu>*, *<networkadapter>*, *<filemonitor>* i *<sampleperiod>*. Unutar elementa *<cpu>* može biti zadana vrijednost *true* ili *false*. Ako je zadana vrijednost *true* onda će Podsustav za mjerenje radnih značajki usluga očitavati zauzeće procesora računala domaćina, a ako je *false* očitavanje se neće izvršavati. Unutar elementa *<networkadapter>* također se upisuje vrijednost *true* ili *false* koja određuje da li će Podsustav za mjerenje radnih značajki usluga očitavati opterećenje mrežne kartice računala domaćina. Unutar elementa *<filemonitor>* također se upisuje *true* ili *false* vrijednost koja određuje da li će Podsustav za mjerenje radnih značajki usluga očitavati veličinu određene datoteke na računalu domaćinu. Ako je upisana vrijednost *true* onda je potrebno odrediti i vrijednost atributa *name* elementa *<filemonitor>*. Vrijednost atributa *name* predstavlja relativnu stazu datoteke čija se veličina želi očitavati. Element *<sampleperiod>* određuje iznos

vremenskog trajanja pauze u milisekundama nakon koje se očitavaju zauzeća računalnih sredstava zadanih prije opisanim elementima.

Sadržajem elementa *<loadgenerator>* i njegovih ugniježđenih elemenata određujemo postavke za upravljanje radom Podsustava za pozivi usluga. Slika 6.6 prikazuje primjer postavki elementa *<loadgenerator>*. Atribut *address*, elementa

```
<loadgenerator address="IP adresa">
        <threads>M</threads>
        <totalduration>T1</totalduration>
        <invocationperiod distributionType="value">T2</invocationperiod>
        <xmlload>naziv datoteke</xmlload>
        <servicesrcaddr>adresa</servicesrcaddr>
        <servicedestaddr>adresa</servicedestaddr>
</loadgenerator>
```

Slika 6.6: Primjer postavki elementa *<loadgenerator>*

<loadgenerator>, sadrži IP adresu računala domaćina Podsustava za poziv usluga kojem pridružujemo definirane postavke za upravljanje radom sustava. Unutar elementa *<loadgenerator>* nalaze se elementi *<threads>*, *<totalduration>*, *<invocationperiod>*, *<xmlload>*, *<servicesrcaddr>* i *<servicedestaddr>*. Element *<threads>* određuje koliko će se dretvi stvoriti unutar Podsustava za poziv usluga koje će istovremeno pozivati primjenske usluge. Element *<totalduration>* određuje ukupno trajanje ispitivanja u milisekundama, vrijednost T_1. Element *<invocationperiod>* određuje vrijeme između dva uzastopna poziva usluge s tim da se vrijeme između dva uzastopna pozive usluge modelira korištenjem odgovarajućeg matematičkog modela. Koji će se model koristiti određuje atribut *distributionType* koji može poprimiti vrijednosti *exp* ili *const*. Vrijednost *exp* upućuje na korištenje eksponencijalne razdiobe za modeliranje vremena između dva uzastopna poziva usluge, dok vrijednost *const* određuje da će vrijeme između dva uzastopna poziva usluge uvijek trajati T_2 milisekundi. S elementom *<xmlload>* moguće je odrediti naziv XML datoteke čiji će se sadržaj koristiti kao argument prilikom poziva metode primjenske usluge. Element *<servicesrcaddr>* određuje adresu računala s kojeg se pozivaju usluge, a element *<servicedestaddr>* adresu računala na kojem se pozivana usluga nalazi.

6.2.3 Rezultati s mjerenim radnim značajkama

Na kraju ispitnog slijeda Upravljački podsustav stvara izlaznu datoteku s mjerenim radnim značajkama. Sadržaj izlazne datoteke s mjerenim radnim značajkama ovisi o postavkama za upravljanje radom sustava koje određuju koliko će se Podsustava za mjerenje radnih značajki usluga koristiti i koja će se računalna sredstva pratiti pri radu sustava. Slika 6.7 prikazuje strukturu izlazne datoteke s mjerenim radnim značajkama.

```
adresa;
cpu; download; upload; total; fileSize;
cpu; download; upload; total; fileSize;
…
cpu; download; upload; total; fileSize;
adresa;
cpu; download; upload; total; fileSize;
cpu; download; upload; total; fileSize;
…
cpu; download; upload; total; fileSize;
…
```

Slika 6.7: Struktura izlazne datoteke s mjerenim radnim značajkama

Rezultati prikupljeni s jednog Podsustava za mjerenje radnih značajki usluga zapisani su u bloku koji počinje adresom računala na kojem se taj podsustav nalazi, a završava početkom sljedećeg bloka. Unutar bloka nalaze se nizovi prikupljenih podataka tj. zapisi o zauzeću računalnih sredstava. Zapis o zauzeću računalnih sredstava prikupljen u jednom trenutku može sadržavati *cpu*, *download*, *upload*, *total* i *fileSize* podatkovne elemente. Element *cpu* predstavlja postotno zauzeće procesora, dakle može poprimiti vrijednosti u intervalu od 0 do 100. Element *download* predstavlja količinu mrežnog prometa koje to računalo prima, u trenutku očitavanja zauzeća računalnih sredstava, izraženu u kilobitima po sekundi [Kb/s]. Element *upload* predstavlja količinu mrežnog prometa koje to računalo šalje, u trenutku očitavanja zauzeća računalnih sredstava, izraženu u kilobitima po sekundi [Kb/s]. Element *total* predstavlja ukupnu količinu mrežnog prometa, u trenutku očitavanja zauzeća računalnih sredstava, izraženu u kilobitima po sekundi [Kb/s]. Element *fileSize* predstavlja veličinu datoteke izraženu u kilooktetima [KB].

Broj zapisa o zauzeću računalnih sredstava unutar jednog bloka ovisi o ukupnom trajanju ispitivanja T_{uk} i trajanju vremenske pauze između dva očitanja zauzeća računalnih sredstava T_{st}. Broj blokova zapisa o zauzeću računalnih

sredstava ovisi o broju Podsustava za mjerenje radnih značajki usluga *N* koji su korišteni tijekom ispitivanja. Veličina izlazne datoteke V_{dat} može se opisati izrazom:

$$V_{dat} = f\left(T_{uk}, T_{st}, N, V_{zap}\right) = \left[N * \left(T_{uk} / T_{st}\right)\right] * V_{zap}$$

gdje V_{zap} predstavlja veličinu jednog zapisa o zauzeću računalnih sredstava. Radi procjene veličine izlazne datoteke s mjerenim radnim značajkama pretpostavljaju se sljedeće veličine V_{zap} iznosi 30 okteta [B], *N* iznosi 2, *T* iznosi 500 milisekundi [ms] i T_{uk} iznosi 2,5 minute [min]. Uvrštavanjem pretpostavljenih vrijednosti u gore navedeni izraz dobiva se kao rezultat veličina izlazne datoteke od 17,5 kilookteta [KB].

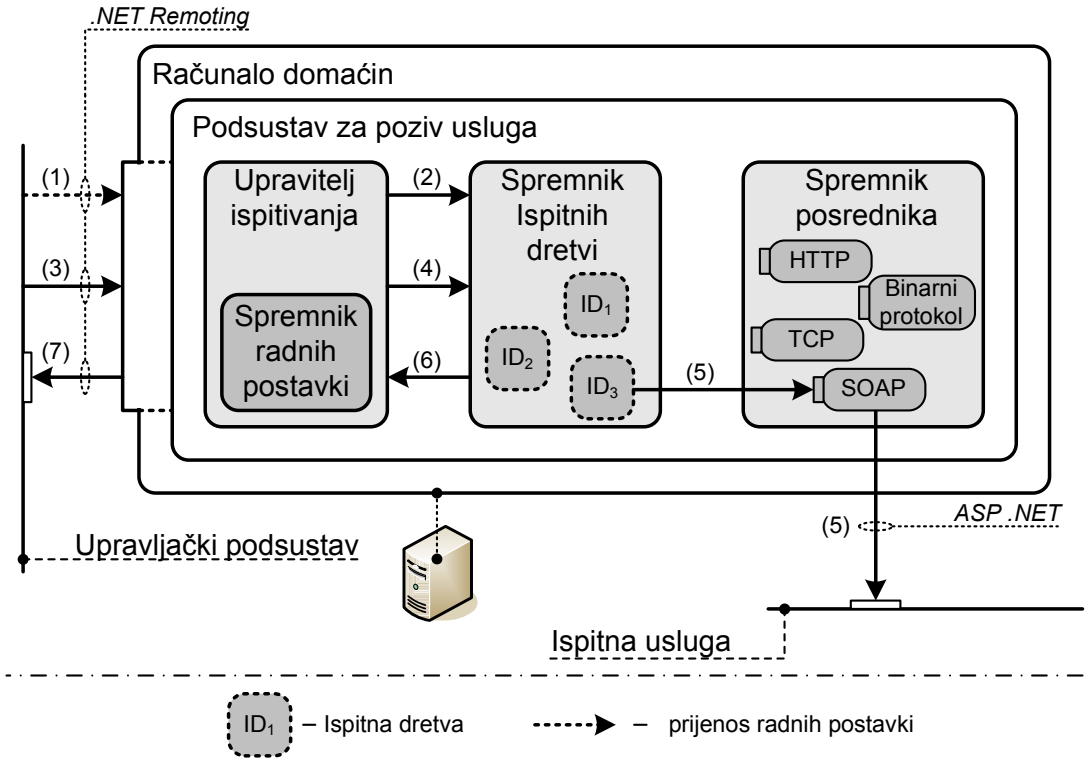
Kao druga mogućnost ostvarenja izlazne datoteke može se koristiti datoteka u XML obliku, ali zbog nedostataka kao što su dodatna potreba za parsiranjem i redundancija zapisa u odnosu na gore navedenu strukturu, ostvarenje datoteka u XML obliku nije korišteno.

6.2.4 Podsustav za poziv usluga

Podsustav za poziv usluga zadužen je za niz operacija stvaranja poziva ispitnih usluga u sustavu zasnovanom na uslugama koje čine ispitni sustav. Podsustav za poziv usluga sastoji se od modula Upravitelj ispitivanja, modula Spremnik radnih postavki, modula Spremnik Ispitnih dretvi i modula Spremnik posrednika.

Modul Upravitelj ispitivanja na osnovi primljenih radnih postavki stvara Ispitne dretve koje pozivaju ispitne usluge. Modul Spremnik radnih postavki služi za spremanje primljenih radnih postavki od Upravljačkog podsustava. Modul Spremnik Ispitnih dretve sprema Ispitne dretve koje stvara modul Upravitelj sustava. Modul Spremnik posrednika sprema posrednike koje Ispitne dretve koriste za poziv ispitnih usluga.

Slika 6.8 prikazuje programsku arhitekturu Podsustava za poziv usluga. Upravljački podsustav pokreće (1) Podsustav za poziv usluga pri čemu se prenose postavke potrebne za upravljanje radom Podsustava za poziv usluga. Upravljački podsustav poziva metodu *SetTestParameters* kojoj predaje postavke za upravljanje radom sustava. Metode sučelja Podsustava za poziv usluga prikazane su u tablici 6.2. Radne postavke pohranjuju se u modul Spremnik radnih postavki. Postavke



Slika 6.8: Programska arhitektura Podsustava za poziv usluga

potrebne za upravljanje radom sustava uključuju radne postavke Podsustava za poziv usluga zadane od strane Upravitelja sustava, XML podatke koji se koriste kao argumenti prilikom poziva primjenske usluge i IP adresu računala domaćina Upravljačkog podsustava. Nakon spremanja radnih postavki modul Upravitelj ispitivanja učitava radne postavke te na osnovi učitanih radnih postavki stvara zadani broj Ispitnih dretvi kojima predaje potrebne radne postavke. Stvorene Ispitne dretve spremaju se (2) u modul Spremnik Ispitnih dretvi. Upravljački podsustav pokreće ispitivanja (3) na Podsustavu za poziv usluga pozivanjem metode *TestStart*. Modul Upravitelj ispitivanja pokreće (4) Ispitne dretve koje počinju s pozivanjem ispitnih usluga. Ispitne dretve, ovisno o spremljenim radnim postavkama, koriste odgovarajući posrednik za poziv ispitnih usluga (5) kojeg dohvaćaju iz modula Spremnik posrednika. Po isteku ukupnog vremena ispitivanja sve Ispitne dretve završavaju sa stvaranjem novih poziva usluga i potom modul Spremnik Ispitnih dretvi šalje poruku (6) o kraju ispitivanja modulu Upravitelj ispitivanja. Po primitku poruke modul Upravitelj ispitivanja šalje (7) poruku Upravljačkom podsustavu o završetku ispitivanja kojom dojavljuje uspješnosti izvođenja ispitivanja.

Ime	Ulazni parametar	Izlazni parametar
SetTestParameters	Radne postavke	*True* ako su radne postavke uspješno učitane, inače *false*
TestStart	-	*True* ako je ispitivanje uspješno počelo, inače *false*

Tablica 6.2: Metode sučelja Podsustava za poziv usluga

Ispitna dretva

Ispitna dretve ostvaruje pozivanje primjenskih usluga čije radne značajke ispitujemo. Pseudokod Ispitne dretve prikazan je na slici 6.9. Nakon pokretanja

```
Tpoc = trenutno_vrijeme();
Tuk = ukupno vrijeme ispitivanja;
dok ( Tuk > (trenutno_vrijeme() – Tpoc) )
{
        pozovi_uslugu;
        odredi trajanje vremenske stanke Tst;
        obustavi_izvođenje(Tst);
}
```

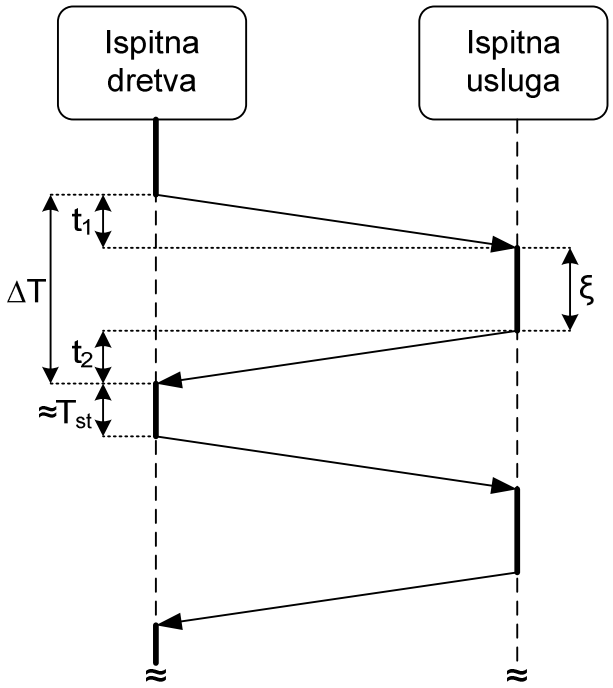
Slika 6.9: Pseudokod Ispitne dretve

Ispitna dretva očitava trenutno vrijeme koje služi za mjerenje isteka ukupnog vremena ispitivanja. Nakon toga Ispitna dretva provjerava da li je isteklo ukupno vrijeme ispitivanja. Ako je odgovor pozitivan Ispitna dretva trajno zaustavlja izvođenje. Ako je odgovor negativan Ispitna dretva sinkrono poziva primjensku uslugu. Sinkroni način pozivanja usluge nalaže da Ispitna dretva privremeno zaustavi izvođenje sve dok poziv primjenske usluge ne završi sa izvođenje i Ispitna dretva primi poruku odgovora. Nakon završetka poziva primjenske usluge Ispitna dretva radi privremenu stanku. Duljina trajanja stanke određena je postavkama za upravljanje radom sustava. Vremenski period između dva poziva primjenske usluge Upravitelj sustava određuje postavljanjem željene vrijednosti elementa *<invocationperiod>* i postavljanjem vrijednosti atributa *distributionType*. Za modeliranje vremena između dva uzastopna poziva usluge Upravitelju sustava na raspolaganju se nalaze dva matematička modela. Prvi model određuje da će vrijeme između dva uzastopna poziva usluge biti jednako vrijednosti parametra *<invocationPeriod>* zadanog s radnim postavkama. Drugi model omogućuje korištenje eksponencijalne razdiobe za

modeliranje vremena između dva uzastopna poziva usluge. Vrijeme između dva poziva usluge određuje se primjenom izraza:

$$\ln\left(\left(1-X\right)*e^{-T}\right)$$

Korištenjem ugrađenog generatora slučajnih brojeva, pri svakoj vremenskoj stanci nakon poziva primjenske usluge, generira se slučajna vrijednost *X* u rasponu [0, 1>. Generirana vrijednost *X* uvrštava se u gore navedenu formulu zajedno s vrijednošću parametra <*invocationperiod*> koji predstavlja vrijednost *T*. Rezultat uvrštavanja predstavlja iznos vremenske stanke između dva uzastopna poziva usluge. Time je osigurano da su vremena privremene obustave izvođenja Ispitne dretve usklađena s eksponencijalnom razdiobom.



t_1 - vrijeme potrebno za slanje zahtjeva ΔT - ukupno trajanje sinkronog poziva

ξ - vrijeme trajanja obrade zahtjeva T_{st} - trajanje vremenske stanke

t_2 - vrijeme potrebno za slanje odgovora

Slika 6.10: Prikaz segmenata vremenskog trajanja poziva Ispitne usluge

Slika 6.10 prikazuje vremenski model pozivanja ispitnih usluga od strane Ispitne dretve. Tijekom poziva ispitne usluge potrebno je prenijeti poruku zahtjeva od računala domaćina Ispitne dretve do računala domaćina ispitne usluge. Vremensko trajanje potrebno za slanje poruke zahtjeva označeno je s t_1. Po primitku zahtjeva na

računalu domaćinu ispitne usluge započinje obrada zahtjeva. Vremensko trajanje potrebno za obradu zahtjeva označeno je s ξ. Nakon završetka obrade dolazi do slanja poruke odgovora od računala domaćina Ispitne usluge do računala domaćina Ispitne dretve. Vremensko trajanje potrebno za slanje poruke odgovora označeno je s t_2. Ukupno trajanje sinkronog poziva usluge označeno je s ΔT i može se izračunati uporabom izraza:

$$\Delta T = t_1 + \xi + t_2$$

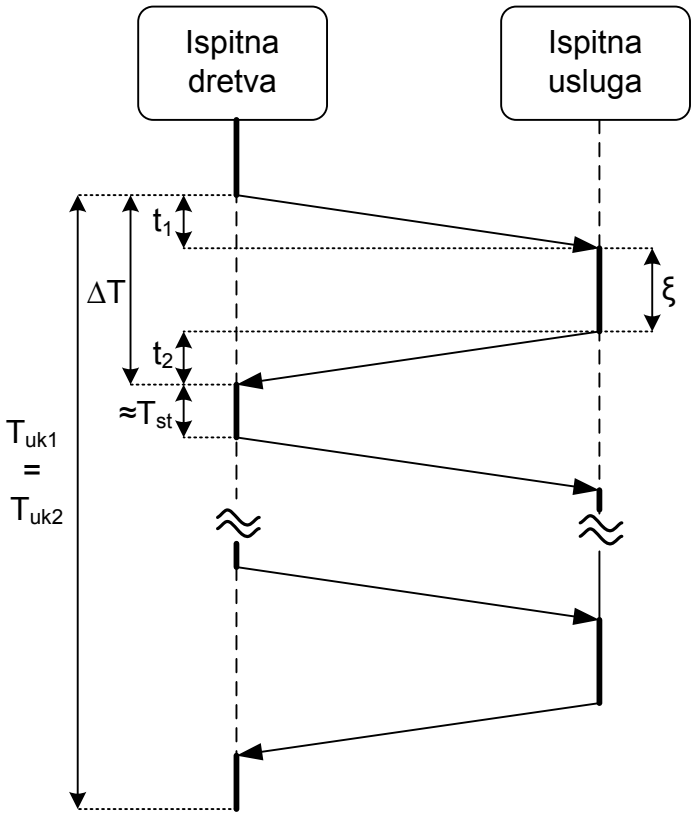
Po primitku poruke odgovora Ispitna dretva računa trajanje vremenske stanke T_{st} i privremeno obustavlja izvođenje za T_{st} vremenskih jedinica. Trajanje postupka računanja iznosa vremenske stanke T_{st} u odnosu na samo trajanje vremenske stanke T_{st} je zanemarivo. Uzimajući u obzir sve navedeno vremensko trajanje cjelokupnog procesa poziva ispitne usluge na strani Ispitne dretve može se izračunati korištenjem izraza:

$$T_{poziva} = \Delta T + T_{st} = t_1 + \xi + t_2 + T_{\mathrm{st}}$$

Neka je vrijednost parametra *<totalDuration>* jednaka T_{uk}. Tada se može za slučaj kada ukupno vremensko trajanje ispitivanja istekne neposredno prije provjere uvjeta trajne obustave rade Ispitne dretve pisati da trajanje T_{uk} iznosi:

$$T_{uk} = N * T_{poziva} = N * \Delta T + N * T_{st}$$

Slika 6.11 prikazuje segmente vremenskog trajanja poziva ispitnih usluga u slučaju isteka ukupnog vremenskog trajanja ispitivanja neposredno prije provjere uvjeta trajne obustave rada Ispitne dretve.



t_1 - vrijeme potrebno za slanje zahtjeva T_{st} - trajanje vremenske stanke

ξ - vrijeme trajanja obrade zahtjeva T_{uk1} - zadano ukupno trajanje ispitivanja

t_2 - vrijeme potrebno za slanje odgovora T_{uk2} - stvarno ukupno trajanje ispitivanja

ΔT - ukupno trajanje sinkronog poziva

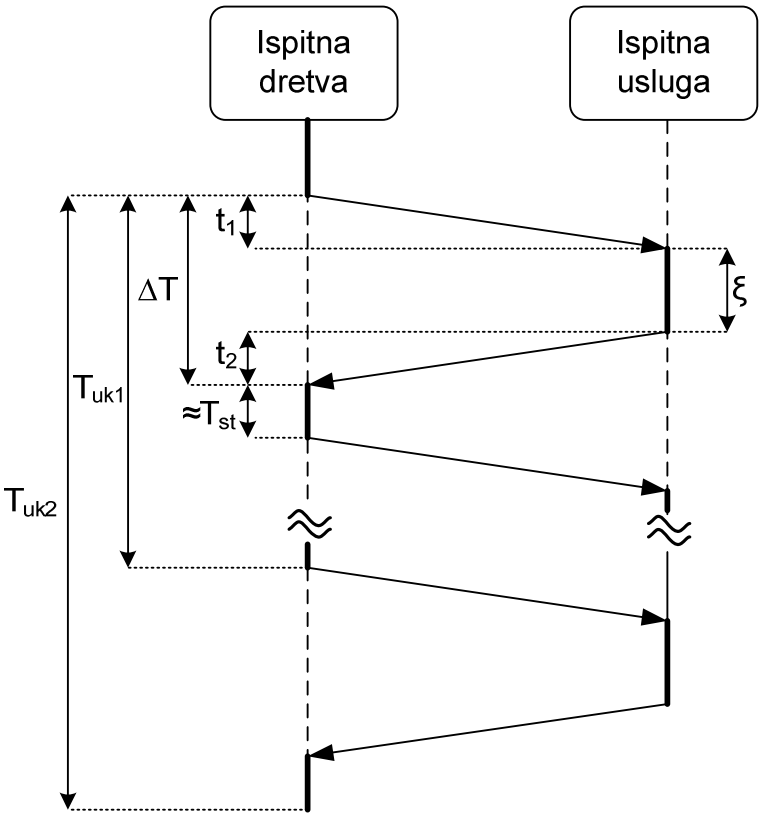
Slika 6.11: prikaz segmenata vremenskog trajanja poziva ispitne usluge u najgorem slučaju

U slučaju da ukupno vremensko trajanje ispitivanja istekne odmah nakon provjere uvjeta trajne obustave rada Ispitne dretve trajanje ispitivanja T_{uk} iznosi:

$$T_{uk} = (N+1) * T_{poziva} = (N+1) * \Delta T + (N+1) * T_{st}$$

Slika 6.12 prikazuje segmente vremenskog trajanja poziva ispitnih usluga u slučaju isteka ukupnog vremenskog trajanja ispitivanja neposredno nakon provjere uvjeta trajne obustave rada Ispitne dretve.

Maksimalno odstupanje od zadanog ukupnog vremenskog trajanja ispitivanja biti će jednako iznosu trajanja cjelokupnog procesa poziva ispitne usluge na strani Ispitne dretve tj. vrijednosti T_{poziva}. Za potrebe izračuna trajanja vrijednosti T_{poziva} pretpostavljaju se sljedeće vrijednosti: veličina poruke zahtjeva iznosi 1 megaoktet [MB] (engl. *megabyte*), veličina poruke odgovora iznosi 1 megaoktet [MB], trajanje



t_1 - vrijeme potrebno za slanje zahtjeva T_{st} - trajanje vremenske stanke

ξ - vrijeme trajanja obrade zahtjeva T_{uk1} - zadano ukupno trajanje ispitivanja

t_2 - vrijeme potrebno za slanje odgovora T_{uk2} - stvarno ukupno trajanje ispitivanja

ΔT - ukupno trajanje sinkronog poziva

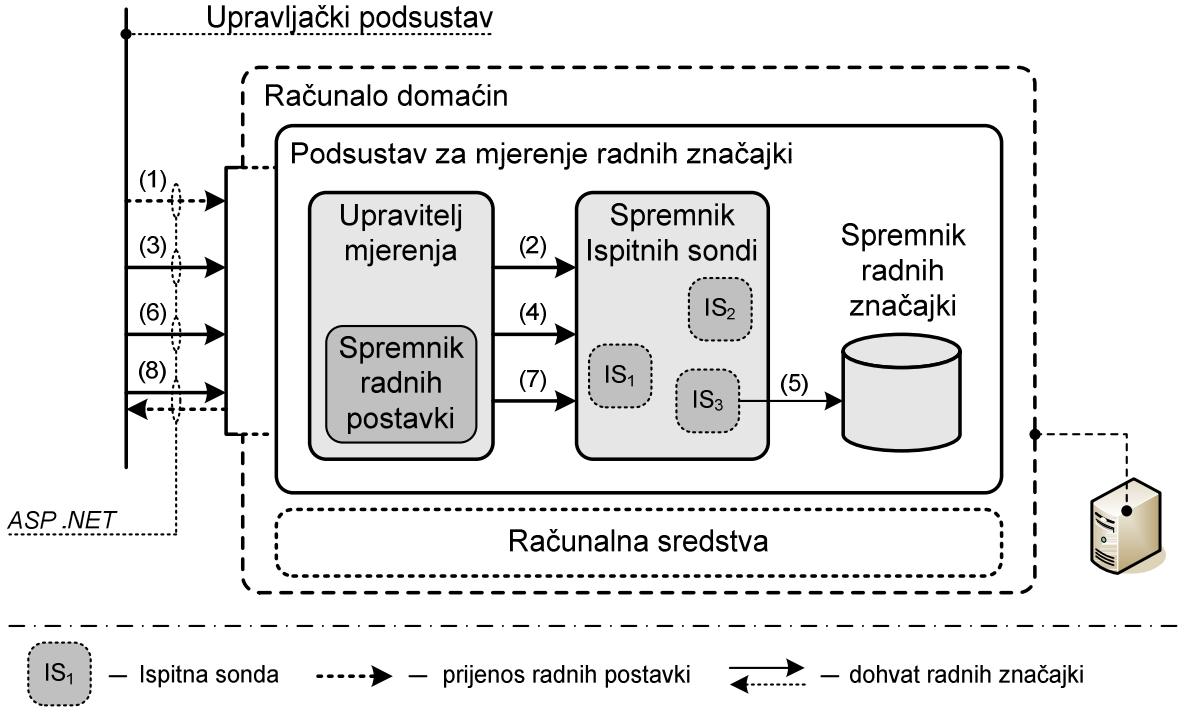
Slika 6.12: Prikaz segmenata vremenskog trajanja poziva ispitne usluge u najgorem slučaju

obrade zahtjeva ispitne usluge iznosi 400 milisekundi [ms], a brzina mreže koja omogućuje komunikaciju računala domaćina Ispitne dretve i računala domaćina ispitnih usluga iznosi 100 megabita po sekundi [Mbps]. Poznavajući brzinu komunikacijske mreže moguće je odrediti trajanje prijenosa poruke zahtjeva i odgovora. Trajanje prijenosa poruke zahtjeva odnosno odgovora jednako je budući da su pretpostavljene veličine jednake i iznosi 10,48 milisekundi [ms]. Za izračun trajanja vrijednosti T_{poziva} potrebno je poznavati još i vrijednost T_{st}. Ako se pretpostavi iznos trajanja stanke između dva uzastopna poziva usluge na vrijednost 1 s tada vrijednost T_{poziva} iznosi približno 1,5 s. Ako je ukupno trajanje ispitivanja veće od 2,5 minute [min], odnosno 150 sekundi [s], tada maksimalno relativno odstupanje iznosi manje od 1%.

6.2.5 Podsustav za mjerenje radnih značajki usluga

Podsustav za mjerenje radnih značajki usluga zadužen je za mjerenje zauzeća računalnih sredstava računala domaćina. Sastoji se od modula Upravitelja mjerenja, modula Spremnik radnih postavki, modula Spremnik Ispitnih sondi i modula Spremnik radnih značajki.

Modul Upravitelja mjerenja na osnovi primljenih radnih postavki stvara Ispitne sonde koje pohranjuje u modul Spremnik Ispitnih sondi te stvorene Ispitne sonde periodički pokreće radi očitavanja zauzeća računalnih sredstava. Modul Spremnik radnih postavki služi za spremanje radnih postavki primljenih od Upravljačkog podsustava, dok modul Spremnik Ispitnih sondi služi za spremanje stvorenih Ispitnih sondi. Modul Spremnik radnih značajki služi za spremanje informacija o zauzeću računalnih sredstava koje su prikupile Ispitne sonde. Programska arhitektura Podsustava za mjerenje radnih značajki usluga prikazana je na slici 6.13 .



Slika 6.13: Ostvarenje Podsustava za mjerenje radnih značajki usluga

Upravljački podsustav pokreće (1) Podsustav za mjerenje radnih značajki usluga pozivom metode *Init* sučelja Podsustava za mjerenje radnih značajki usluga. Prilikom poziva metode *Init* kao ulazni argument predaju se radne postavke Podsustava za mjerenje radnih značajki usluga. Metode sučelja Podsustava za

mjerenje radnih značajki prikazane su u tablici 6.3. Radne postavke pohranjuju se u modul Spremnik radnih postavki. Po primitku radnih postavki modul Upravitelj mjerenja stvara potrebne Ispitne sonde koje zatim sprema (2) u modul Spremnik ispitnih sondi. (3) Upravljački podsustav šalje poruku za početak mjerenja pozivom metode *StartMonitoring*. Modul Upravitelj mjerenja pokreće (4) Ispitne sonde koje se nalaze u modulu Spremnik Ispitnih sondi. Modul Upravitelj mjerenja periodično pokreće (4) Ispitne sonde radi očitavanja zauzeća onog računalnog sredstva računala domaćina za koje su određene. Očitane vrijednosti pohranjuju se u modul Spremnik radnih značajki. (6) Upravljački podsustav šalje poruke o završetku mjerenja, pozivom metode *StopMonitoring,* nakon koje modul Upravitelj mjerenja prazni (7) modul Spremnik Ispitnih sondi. Upravljački podsustav šalje zahtjev za dohvatom (8) mjerenih radnih značajki, pozivom metode *GetStatistics*, nakon kojeg dolazi do prijenosa svih podataka spremljenih u modulu Spremnik radnih značajki.

Ime	Ulazni parametar	Izlazni parametar
Init	Radne postavke	*True* ako su radne postavke uspješno učitane, inače *false*
StartMonitoring	-	-
StopMonitoring	-	-
GetStatistics	-	*Skup prikupljenih mjerenih radnih značajki*

Tablica 6.3: Metode sučelja Podsustava za mjerenje radnih značajki usluga

Podsustav za mjerenje radnih značajki usluga ostvaren je kao zaseban razred kojim je zatim proširen Podsustav za lokalno nadgledanje Sustava za nadgledanje korisnika i rada usluga tako da se Podsustavu za mjerenje radnih značajki usluga pristupa preko sučelja Podsustava za lokalno nadgledanje.

Ispitne sonde

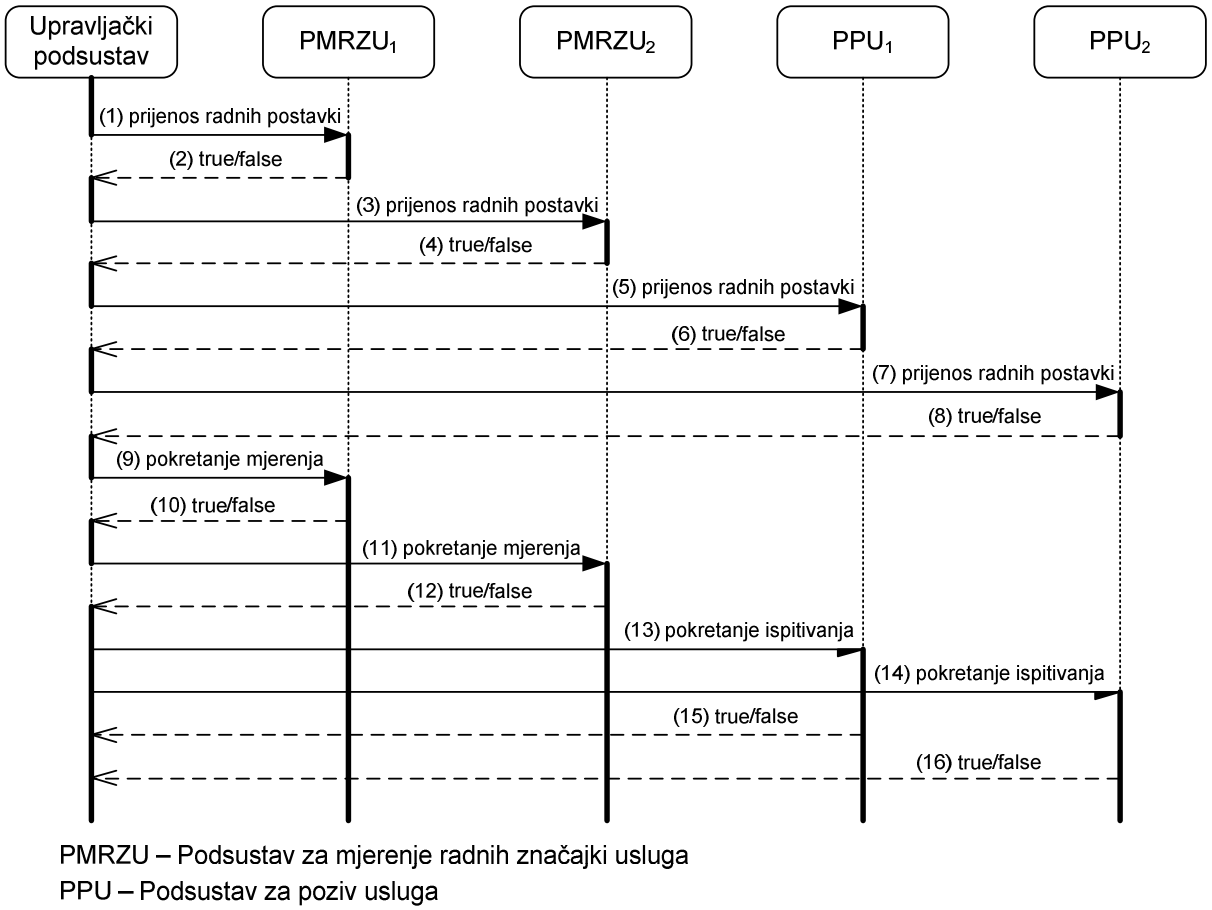
Ispitne sonde zadužene su za očitavanje stanja o zauzeću određenog računalnog sredstva računala domaćina. Ostvarene su ispitne sonde za praćenje mrežnog prometa, za praćenje zauzeća procesora i za praćenje veličine datoteke.

Ispitna sonda za praćenje mrežnog prometa očitava promet na svim raspoloživim mrežnim sučeljima računala domaćina. Podaci koje očitava su iznos mrežnog prometa prema računalu, iznos mrežnog prometa od računala te ukupni iznos mrežnog prometa odnosno promet u oba smjera. Svi očitani podaci izraženi su u kilobitima po sekundi [Kb/s]. Ispitna sonda za praćenje zauzeća procesora očitava postotno zauzeće procesora, dakle očitane vrijednosti su iz intervala od 0 do 100. Ispitna sonda za praćenje veličine datoteke očitava trenutnu veličinu datoteku. Izbor datoteke tj. koja se datoteka prati zadaje se s postavkama za upravljanje radom Podsustava za mjerenje radnih značajki usluga. Očitana veličina datoteka izražena je u kilooktetima [KB].

6.2.6 Upravljački protokol sustava

Upravljački protokol sustava prikazuje međudjelovanje podsustava Sustava za ispitivanje radnih značajki usluga. Upravljački protokol dijeli se na tri koraka: priprema provedbe ispitivanja usluga, provedba ispitivanja usluga i završetak ispitivanja usluga. Tijekom pripreme provedbe ispitivanja usluga pokreću se svi podsustavi i prenose se radne postavke svim potrebnim podsustavima. Sve do isteka ukupnog vremena ispitivanja traje provedba ispitivanja usluga tijekom koje se pozivaju ispitne usluga. Završetak ispitivanja usluga podrazumijeva zaustavljanje rada svih podsustava i prikupljanje statističke podatke. Tri koraka upravljačkog protokola prikazana su redom na slikama 6.14, 6.15 i 6.16 .

Slika 6.14 prikazuje pripremu provedbe ispitivanja usluga. Nakon pokretanja Upravljačkog podsustava od strane Upravitelja sustava, Upravljački podsustav prenosi (1 - 4) radne postavke Podsustavima za mjerenje radnih značajki usluga. Na slici su radi jednostavnosti prikaza prikazana samo dva Podsustava za mjerenje radnih značajki usluga. Nakon toga dolazi do prijenosa radnih postavki Podsustavima za poziv usluga (5 - 8). Na slici su radi jednostavnosti prikaza prikazana samo dva Podsustava za mjerenje radnih značajki. Zatim dolazi do pokretanja mjerenja (9 - 12) na Podsustavima za mjerenje radnih značajki usluge te do pokretanja ispitivanja (13 - 16) na Podsustavima za poziv usluga. Nakon primitka poruke za početak ispitivanja Podsustavi za poziv usluga počinju pozivati ispitne usluge. Pokretanja ispitivanja na Podsustavima za poziv usluga ostvareno je asinkronim pozivom. Asinkroni poziv nalaže da pozivajuća dretva pozove metodu i nastavi sa daljnjim izvođenjem.

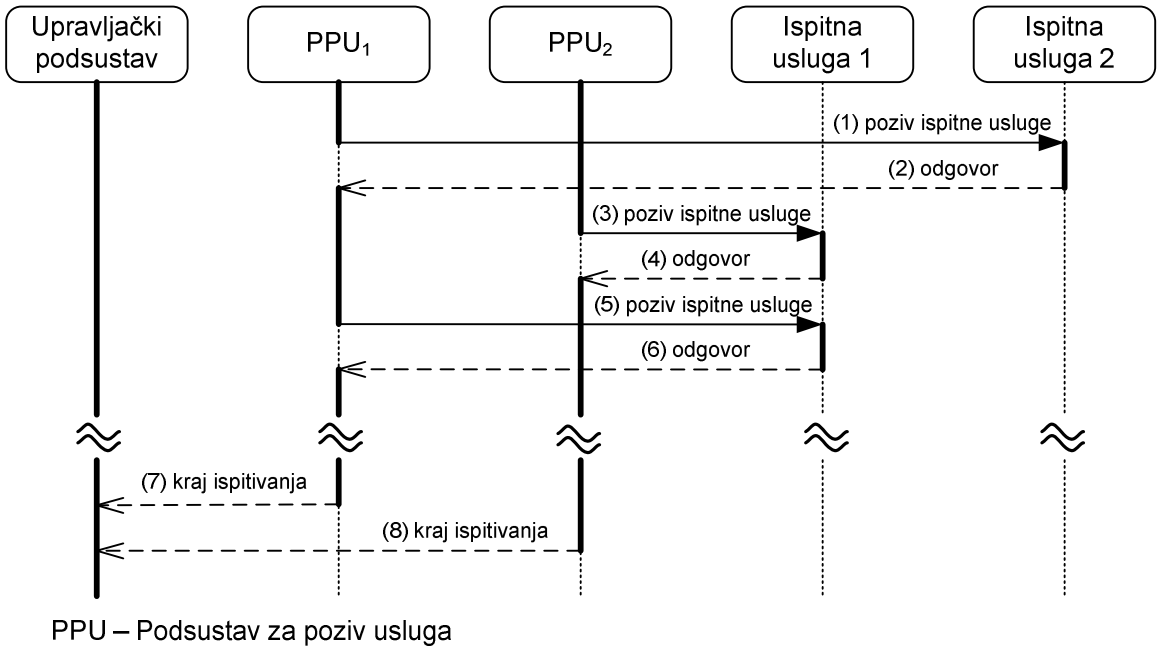


PMRZU – Podsustav za mjerenje radnih značajki usluga
PPU – Podsustav za poziv usluga

Slika 6.14: Priprema provedbe ispitivanja usluga

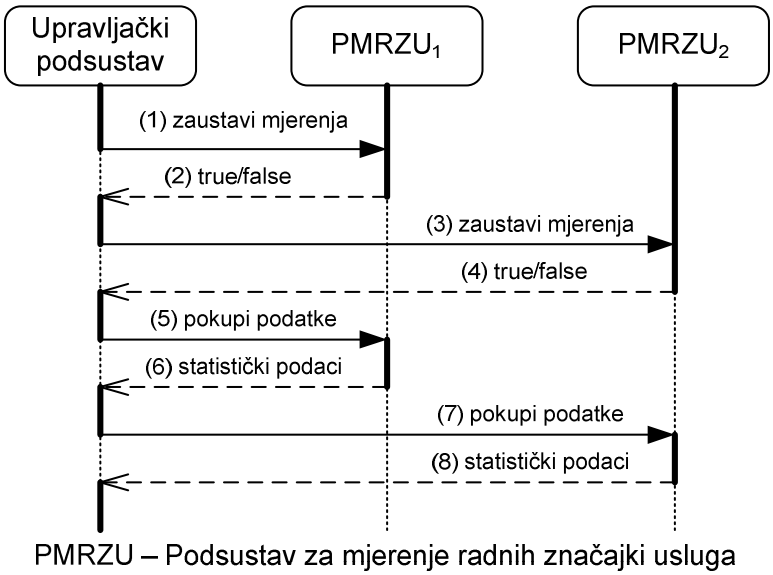
Pozivajuća dretva mora imati određeni dio programskog koda koji obrađuje kraj izvršavanja pozvane metode. Asinkronim pozivima moguće je ostvariti veći broj poziva metoda u nekom vremenskom periodu u odnosu na sinkrone pozive. Budući da Podsustavi za poziv usluga trebaju početi s pozivanjem usluga istovremeno primijenjeni su asinkroni pozivi za pokretanje ispitivanja.

Slika 6.15 prikazuje provedbu ispitivanja usluga. Podsustavi za poziv usluga pozivaju (1 - 6) ispitne usluge primjenom komunikacijskih protokola određenih radnim postavkama Podsustava za poziv usluga. Na slici su radi jednostavnosti prikaza prikazane samo dvije ispitne usluge. Podsustavi za poziv usluga pozivaju ispitne usluge sinkrono sve do isteka ukupnog vremena ispitivanja. Po isteku ukupnog vremena ispitivanja Podsustavi za poziv usluga javljaju kraj ispitivanja (7 - 8) Upravljačkom podsustavu. Nakon što svi Podsustavi za poziv usluga jave kraj ispitivanja nastupa završetak provedbe ispitivanja usluga.



Slika 6.15: Provedba ispitivanja usluga

Slika 6.16 prikazuje završetak ispitivanja usluga. Po završetku provedbe ispitivanja usluga Upravljački podsustav zna da su sva ispitivanja završila stoga zaustavlja mjerenja (1 - 4) na Podsustavima za mjerenje radnih značajki usluga. Nakon toga Upravljački podsustav prikuplja (5 - 8) sve podatke o mjerenim radnim značajkama te s tim završava posljednji korak upravljačkog protokola.



Slika 6.16: Završetak ispitivanja usluga

7 Ispitivanje radnih značajki Sustava za nadgledanje rada

S ciljem ispitivanja ispravnosti i prikaza uporabljivosti Sustava za ispitivanje radnih značajki usluga ostvareni sustav prilagođen je za rad unutar PIE programske okoline. U tu svrhu provedena su ispitivanja radnih značajki PIE podsustava Sustav za nadgledanje rada korisnika i usluga. Tijekom ispitivanja Sustav za ispitivanje radnih značajki usluga stvara opterećenje unutar PIE okoline. Stvoreno opterećenje uzrokuje zauzimanja računalnih sredstava računala domaćina Sustava za nadgledanje rada korisnika i usluga. Sustav za ispitivanje radnih značajki usluga očitava zauzeće računalnih sredstava. Na temelju stvorenih zapisa o zauzeću računalnih sredstava moguće je vrednovati rad Sustava za nadgledanje rada korisnika i usluga.

U ovisnosti o korištenom protokolu razmjene podataka o korištenju usluga, Sustav za nadgledanje rada korisnika i usluga u različitoj mjeri će zauzimati sredstva računala domaćina. S ciljem mjerenja radnih značajki i vrednovanja rada Sustava za nadgledanje rada korisnika i usluga i njegovih protokola razmjene podataka provedena su mjerenja zauzeća sljedećih računalnih sredstava: mrežne propusnosti, diskovnog prostora i procesorskog vremena računala domaćina primjenom Sustava za ispitivanje radnih značajki usluga.

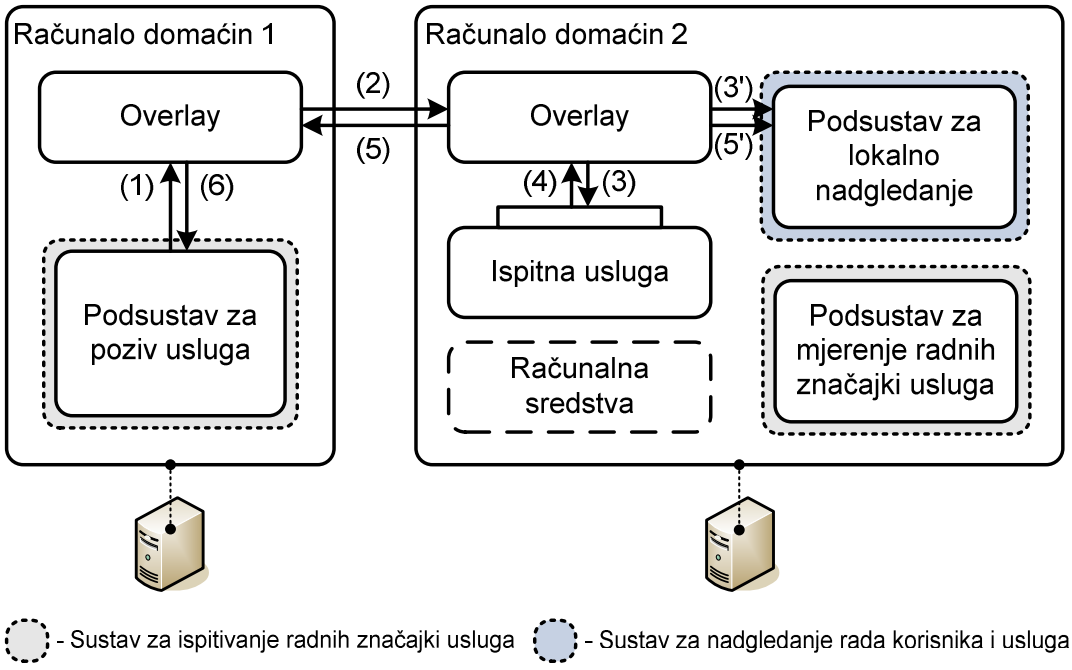
U odjeljku 7.1 opisan je ostvareni ispitni sustav. U odjeljku 7.2 opisane su Ispitna okolina i radne postavke koje su korištene za ispitivanje Sustava za nadgledanje rada korisnika i usluga, dok su u odjeljku 7.3 prikazani i opisani rezultati provedenog ispitivanja.

7.1 Objedinjavanje Ispitnog sustava sa Sustavom za nadgledanje rada

Za potrebe mjerenja radnih značajki i vrednovanja rada Sustava za nadgledanje rada korisnika i usluga i njegovih protokola razmjene podataka ostvaren je ispitni sustav koji se sastoji od Sustava za ispitivanje radnih značajki usluga, Sustava za nadgledanje rada korisnika i usluga i Ispitne usluge.

Ispitna usluga obradom pristiglih poruka zahtjeva stvara opterećenje procesora računala domaćina primjenom petlje radnog čekanja. Ispitna usluga je prijavljena za nadgledanje na Podsustavu za lokalno nadgledanje. Budući da je

Ispitna usluga prijavljena za nadgledanje, Podsustav za lokalno nadgledanje presreće poruke zahtjeva i odgovora Ispitne usluge, analizira sadržaj poruka i izdvaja sadržaj poruka. Postupak presretanja poruka zahtjeva i odgovora, analize i izdvajanja sadržaja poruka troši procesorsko vrijeme računala domaćina. Izdvajanje sadržaja poruka zahtjeva i odgovora Ispitnih usluga Podsustav za lokalno nadgledanje stvara zapise o korištenju Ispitne usluge. Stvoreni zapisi spremaju se lokalno ili u središnji spremnik na računalu gdje je smješten Podsustav za globalno nadgledanje. Ako se zapisi spremaju lokalno tada se zauzima lokalni diskovni prostor. Ako se spremaju u središnji spremnik tada je zapise potrebno prenijeti putem računalne mreže do računala na kojem se nalazi Podsustav za globalno nadgledanje i središnji spremnik što uzrokuje povećanje mrežnog prometa od računala domaćina. Sustav za ispitivanje radnih značajki usluga pozivima Ispitne usluge stvara opterećenje u ispitnom sustavu. Pozivi Ispitne usluge tako uzrokuju zauzeće računalnih sredstava, računala domaćina ispitnog sustava, od strane Sustava za nadgledanje korisnika i usluga i Ispitne usluge koje Sustav za ispitivanje radnih značajki usluga mjeri.



Slika 7.1: Osnova rada ostvarenog Ispitnog sustava

Slika 7.1 prikazuje osnovu rada ostvarenog Ispitnog sustava. Podsustav za poziv usluga poziva Ispitnu uslugu. Poruka zahtjeva šalje se lokalnom sustavu Overlay (1). Lokalni sustav Overlay korištenjem tablica prosljeđivanja prosljeđuje (2)

poruku zahtjeva Overlay sustavu na računalu domaćinu 2. Primanje poruke zahtjeva na računalu domaćinu 2 stvara opterećenje na mrežnoj kartici. Sustav Overlay na računalu domaćinu 2 prosljeđuje (3) primljenu poruku zahtjeva Ispitnoj usluzi. Budući da je Ispitna usluga prijavljena za nadgledanje, Podsustav za lokalno nadgledanje presreće (3') poruku zahtjeva Ispitne usluge. Podsustav za lokalno nadgledanje analizira i izdvaja sadržaj zahtjeva. Analiza i izdvajanje sadržaja zahtjeva troši procesorsko vrijeme računala domaćina 2. Obrada poruke zahtjeva od strane Ispitne usluge također troši procesorsko vrijeme računala domaćina 2. Ispitna usluga obrađuje poruku zahtjeva i stvara poruku odgovora koju šalje (4) lokalnom sustavu Overlay. Sustav Overlay, na računalu domaćinu 2, prosljeđuje (5) primljenu poruka odgovora sustavu Overlay na računalu domaćinu 1. Slanje poruke s računala domaćina 2 do računala domaćina 1 stvara opterećenje na mrežnoj kartici računala domaćina 2. Podsustav za lokalno nadgledanje presreće (5') poruku odgovora Ispitne usluge čiji sadržaj analizira i izdvaja. Po primitku poruke odgovora sustav Overlay, na računalu domaćinu 1, prosljeđuje (6) Podsustavu za poziv usluga poruku odgovora. Cijelo vrijeme tijekom trajanja poziva Ispitne usluge (1 - 6) Podsustav za mjerenje radnih značajki stvara zapise o zauzeću računalnih sredstava.

7.1.1 Ispitna usluga

Ispitna usluga je primjenska usluga koja zauzima procesorsko vrijeme računala domaćina. Sučelje Ispitne usluge sadrži metodu *DoWork* opisanu u tablici 7.1. Prilikom poziva metode *DoWork* kao ulazni argument predaje se podatak tipa *XmlElement*. Metoda kao izlazni argument vraća isti primljeni ulazni argument. Tijelo metode sadrži radnu petlju koja zauzima procesorsko vrijeme računala domaćina. Prosječno trajanje izvođenja Ispitne usluge u radnoj petlji iznosi 370 milisekundi [ms].

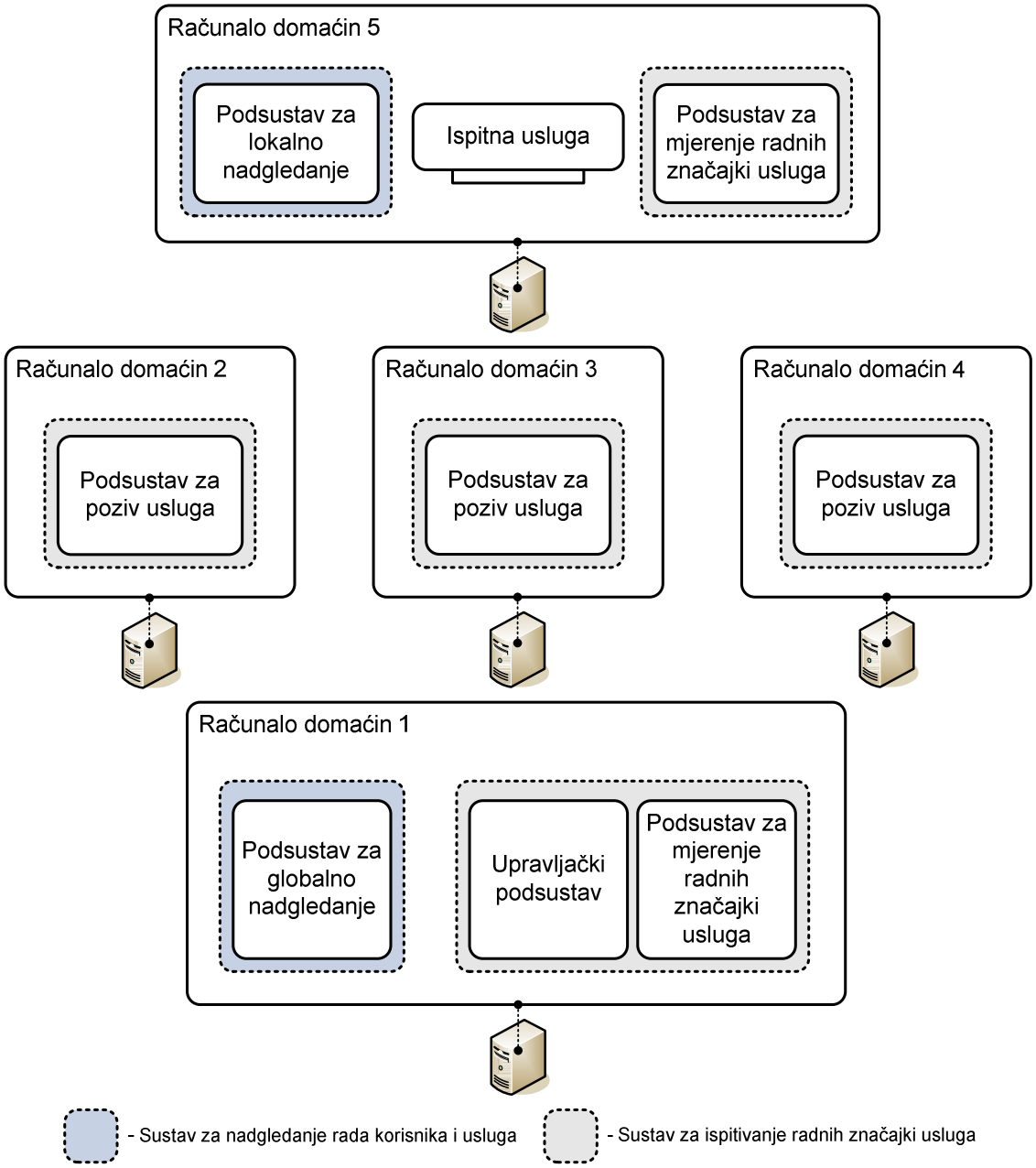
Ime	Ulazni parametar	Izlazni parametar
DoWork	XmlElement	XmlElement

Tablica 7.1: Sučelje Ispitne usluge

7.2 Okolina i postavke za provođenje ispitivanja

Ispitna okolina sastoji se od 5 računala s Windows 2003 Server operacijskim sustavom. Na svakom od računala postavljeni su *.NET* radni okvir i Programibilna

Internet Okolina. Razmještaj komponenti ispitnog sustava u korištenoj ispitnoj okolini prikazan je na slici 7.2.



Slika 7.2: Razmještaj komponenti ispitnog sustava u korištenoj ispitnoj okolini

Na računalu domaćinu 1 nalaze se Podsustav za globalno nadgledanje, Upravljački podsustav i Podsustav za mjerenje radnih značajki usluga. Na računalima domaćinima 2, 3 i 4 nalazi se Podsustav za poziv usluga. Na računalu domaćinu 5 nalaze se Podsustav za lokalno nadgledanje, Podsustav za mjerenje radnih značajki usluga i Ispitna usluga.

Radne postavke potrebne za ispitivanje uključuju postavke Sustava za nadgledanje rada korisnika i usluga i radne postavke Sustava za ispitivanje radnih značajki usluga. Postavke Sustava za nadgledanje rada korisnika i usluga obuhvaćaju postupak prijavljivanja Ispitne usluge za nadgledanje rada na Podsustavu za lokalno nadgledanje. Prijava Ispitne usluge sastoji se od obrasca za nadgledanje usluge. Slika 7.3 prikazuje izgrađeni obrazac za nadgledanje Ispitne usluge. Podsustav za lokalno nadgledanje na osnovi izgrađenog obrasca za

```
<ServiceAccountFormat>
  <TestService>
    <DoWork>
      <request>
        <body>
          <trackPath>
            path:DoWork/load/xmlload/test/test2
          </trackPath>
        </body>
      </request>
      <response>
        <body>
          <trackPath>
            path:DoWorkResponse/DoWorkResult/xmlload/test/test5
          </trackPath>
        </body>
      </response>
    </DoWork>
  </TestService>
</ServiceAccountFormat>
```

Slika 7.3: Obrazac za nadgledanje ispitne usluge

nadgledanje izdvaja elemente *<test2>* iz sadržaja poruka zahtjeva i *<test5>* iz sadržaja poruka odgovora. Elementi *<test2>* i *<test5>* su dio ulaznog i izlaznog argumenta metode *DoWork* Ispitne usluge. Ulazni argument određuje se XML datotekom čije se ime zadaje s radnim postavkama Podsustava za poziv usluga. Struktura korištene XML datoteke prikazana je na slici 7.4. Korijenski element je *<xmlload>* unutar kojeg se nalazi konačan broj *<test>* elemenata. Unutar *<test>* elementa postoji 8 ugniježđenih elemenata *<test1>*, *<test2>*, *<test3>*, *<test4>*, *<test5>*, *<test6>*, *<test7>* i *<test8>* unutar kojih se nalazi niz od 64 slučajno odabrana znaka. Ponavljanjem elementa *<test>* konačan broj puta izgrađene su dvije datoteke veličina 10 kilookteta [KB] i 1000 kilookteta [KB]. Korištenjem priloženog obrasca za nadgledanje Ispitne usluge Podsustav za lokalno nadgledanje izdvajat će približno 1/8 sadržaja poruke zahtjeve i 1/8 sadržaja poruke odgovora Ispitne usluge.

```
<xmlload>
  <test>
       <test1>...</test1>
       <test2>...</test2>
       <test3>...</test3>
       <test4>...</test4>
       <test5>...</test5>
       <test6>...</test6>
       <test7>...</test7>
       <test8>...</test8>
  </test>
  ...
</xmlload>
```

Slika 7.4: Struktura ulaznog argumenta metode DoWork Ispitne usluge

Radne postavke Sustava za ispitivanje radnih značajki zadaju se datotekom s radnim postavkama. Za potrebe ispitivanja i usporedbe protokola za razmjenu podataka Sustava za nadgledanje rada i korisnika usluga izgrađeno je 12 datoteka s radnim postavkama. Izgrađeno je 12 datoteka budući da su se ispitivanja provodila za različite vrijednosti elemenata <*invocationperiod*> i <*xmlload*>. Navedeni elementi i vrijednosti koje poprimaju prikazani su u tablici 7.2. Ispitivanja su izvršena za 6

Element	Vrijednosti
<*invocationperiod*>	100 ms, 1000 ms, 2500ms, 5000ms, 15000ms, 30000ms
<*xmlload*>	load10k.xml, load1000k.xml

Tablica 7.2: Elementi radni postavki koji su mijenjani i njihove vrijednosti

vrijednosti trajanja pauze između dva uzastopna poziva usluge i za 2 veličine ulaznog argumenta metode DoWork Ispitne usluge. Za svaku od datoteka s radnim postavkama napravljeno je po 5 mjerenja. Budući da su mjerenja rađena za 2 protokola razmjene podataka Sustava za nadgledanje rada korisnika i usluga izvršeno je ukupno 120 mjerenja. Elementi radnih postavki koji se nisu mijenjali su <*testrepetitions*>, <*threads*>, <*sampleperiod*> i atribut *distributionType* elementa <*invocationperiod*>. Tablica 7.3 prikazuje elemente radnih postavki koji se nisu mijenjali i njihove vrijednosti.

Element	Vrijednosti
<testrepetitions>	5
<threads>	3
<sampleperiod>	500
distributionType	exp

Tablica 7.3: Elementi radni postavki koji nisu mijenjani i njihove vrijednosti

Tijekom ispitivanja mjerilo se zauzeće računalnih sredstava na računalima domaćinima 1 i 5. Ostvareno je mjerenje zauzeća procesorskog vremena, mrežni promet u oba smjer te u ovisnosti o korištenom protokolu razmjene podataka Sustava za nadgledanje rada korisnika i usluga veličina odgovarajuće datoteke. Ako se koristio protokol za istiskivanje onda se promatrala veličina datoteke, koja je služila kao lokalni spremnik Podsustava za lokalno nadgledanje, na računalu domaćinu 5. Ako se koristio protokol za potiskivanja onda se promatrala veličina datoteke, koja je služila kao središnji spremnik Podsustava za globalno nadgledanje, na računalu domaćinu 1. Slika 7.5 prikazuje jednu od ispitnih datoteka s radnim postavkama.

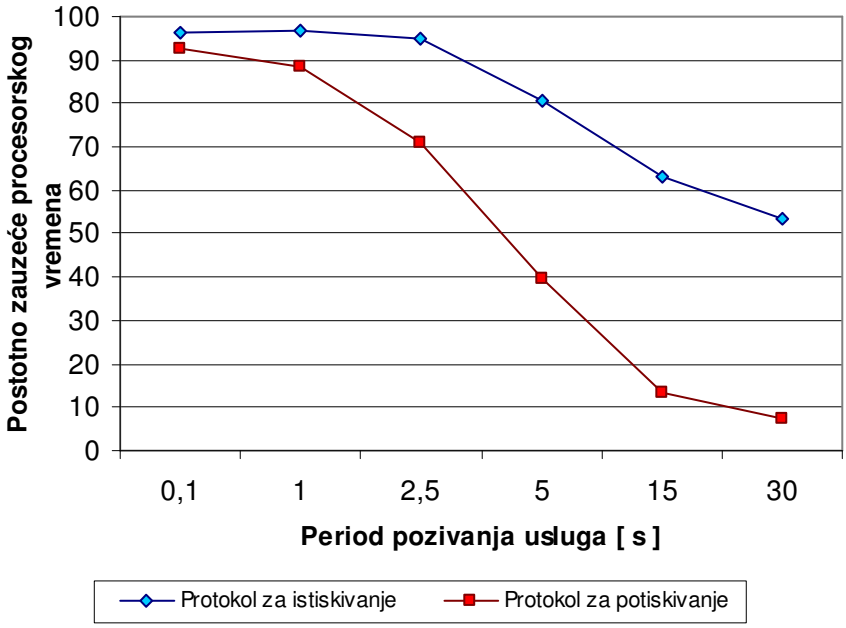
```
<config>
  <tcaddress>adresa računala domaćina 1</tcaddress>
  <testrepetitions>5</testrepetitions>

  <pemconfigurations>
     <pemconfiguration address="adresa računala domaćina 5">
        <cpu>true</cpu>
          <networkadapter>true</networkadapter>
        <filemonitor name="relativna staza">true</filemonitor>
        <sampleperiod>500</sampleperiod>
     </pemconfiguration>

  </pemconfigurations>

  <loadgenerators opmode="overlay">
     <loadgenerator address="adresa računala domaćina 2">
        <threads>3</threads>
        <totalduration>180000</totalduration>
        <invocationperiod distributionType="exp">1000</invocationperiod>
        <xmlload>load10k.xml</xmlload>
          <servicesrcaddr>adresa računala domaćina 2</servicesrcaddr>
          <servicedestaddr>adresa računala domaćina 5</servicedestaddr>
     </loadgenerator>

     <loadgenerator address="adresa računala domaćina 3">
        <threads>3</threads>
        <totalduration>180000</totalduration>
        <invocationperiod distributionType="exp">1000</invocationperiod>
        <xmlload>load10k.xml</xmlload>
          <servicesrcaddr>adresa računala domaćina 3</servicesrcaddr>
          <servicedestaddr>adresa računala domaćina 5</servicedestaddr>
     </loadgenerator>

     <loadgenerator address="adresa računala domaćina 4">
        <threads>3</threads>
        <totalduration>180000</totalduration>
        <invocationperiod distributionType="exp">1000</invocationperiod>
        <xmlload>load10k.xml</xmlload>
          <servicesrcaddr>adresa računala domaćina 4</servicesrcaddr>
          <servicedestaddr>adresa računala domaćina 5</servicedestaddr>
     </loadgenerator>
   </loadgenerators>
</config>
```

Slika 7.5: Primjer ispitne datoteke s radnim postavkama

7.3 Rezultati ispitivanja značajki sustava

Rezultati usporedbe protokola razmjene podataka Sustava za nadgledanje rada korisnika i usluga dijele se u dvije skupine ovisno o vrijednosti veličine korištenog ulaznog argumenta metode *DoWork* Ispitne usluge. U jednoj skupini veličina ulaznog argumenta iznosi 10 kilookteta [KB], a u drugoj 1000 kilookteta [KB]. Svaka skupina sadrži 5 grafova. Grafovi prikazuju iskorištenost pojedinog računalnog sredstva u ovisnosti o vremenskom razmaku između dva uzastopna poziva ispitne usluge *T*. Ukupno trajanje pojedinačnog ispitivanja u obje skupine iznosi 3 minute

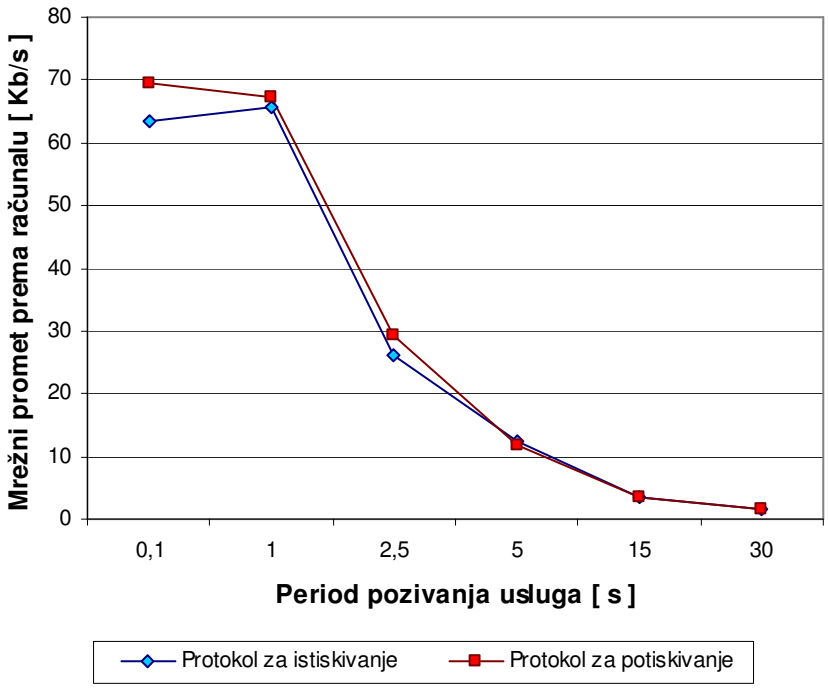
[min]. U nastavku su prikazani grafovi skupine s veličinom ulaznog argumenta iznosa 10 kilookteta [KB].



Slika 7.6: Procesorsko zauzeće u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

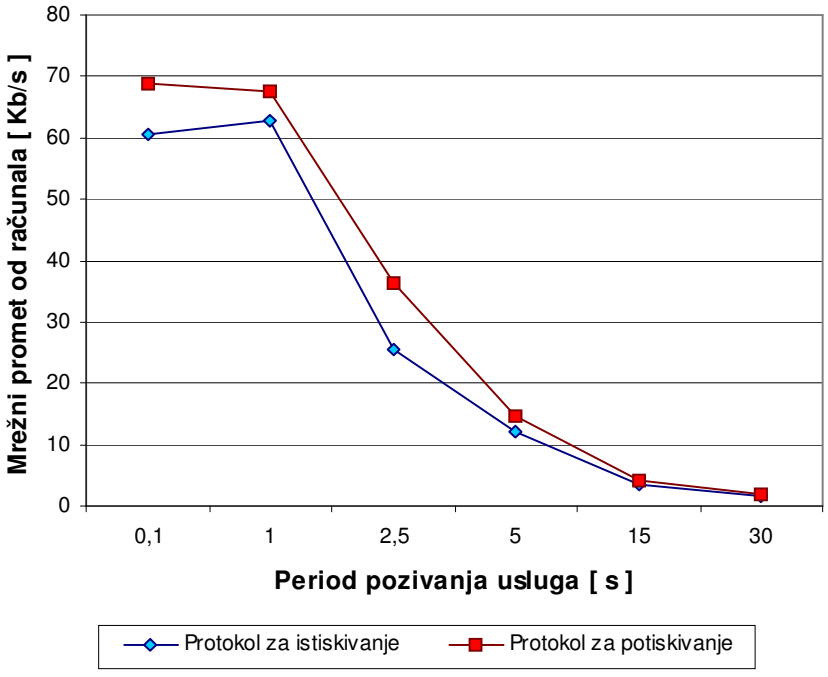
Slika 7.6 prikazuje postotno zauzeće procesorskog vremena na računalu domaćinu 1 u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*. Pri vrijednostima *T* manjima od 1 sekunde [s] razlika između dva protokola za razmjenu podataka u postotnom zauzeća procesora je manja od 10%. Kako vrijednost *T* raste tako raste i razlika u postotnom zauzeća. Pri vrijednostima *T* većima od 15 sekundi [s] razlika u postotnom zauzeću procesora veća je od 50%. Uz korištenje već prikazanog obrasca za nadgledanje Ispitne usluge Podsustav za lokalno nadgledanje pri svakom pozivu Ispitne usluge, uz veličinu ulaznog argumenta od 10 kilookteta [KB], stvara dva zapisa veličine 1,25 kilookteta [KB]. Na osnovi grafa može se zaključiti da operacija zapisivanja u lokalni spremnik, kada je veličina zapisa korištenja usluge relativno mala, zahtjeva više procesorskog vremena u odnosu na operaciju slanja zapisa o korištenju usluge putem računalne mreže.

Slika 7.7 prikazuje mrežni promet prema računalu na računalu domaćinu 1 u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*. Pri vrijednostima *T* manjima od 1 sekunde [s] mrežni promet prema računalu protokola



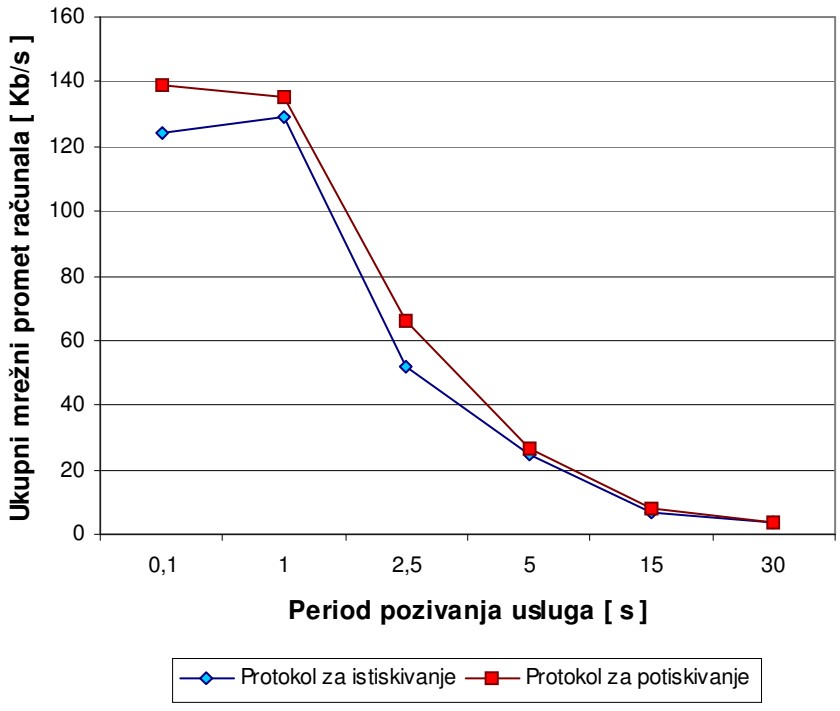
Slika 7.7: Mrežni promet prema računalu u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

za istiskivanje je manji nego kod protokola za potiskivanje. Za vrijednosti T veće i jednake 1 sekundi [s] mrežni promet prema računalu gotovo je identičan što je u skladu s očekivanjima.



Slika 7.8: Mrežni od računala u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

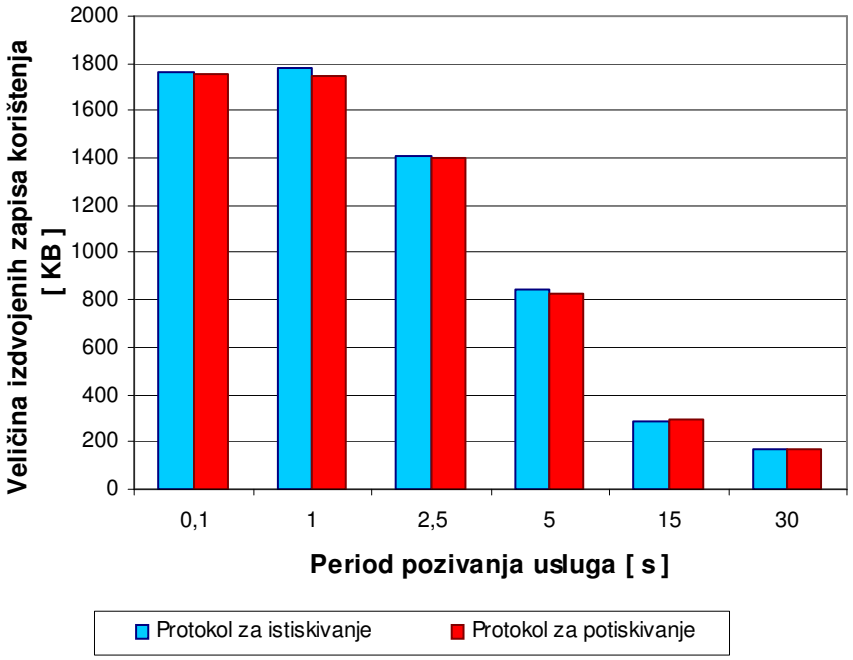
Slika 7.8 prikazuje mrežni promet od računalu računala domaćina 1 u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*. Pri vrijednostima *T* manjima od 15 sekundi [s] mrežni promet od računala protokola za potiskivanje je veći u odnosu na protokol za istiskivanje. Za vrijednosti *T* veće od 15 sekundi mrežni promet od računala se izjednačava. Razlika protokola za razmjenu podataka u iznosu mrežnog prometa od računala je rezultat slanja zapisa o korištenju Ispitne usluge u središnji spremnik. Kako se povećava vremenski razmak između dva uzastopna poziva usluge, tako se smanjuje broj poziva Ispitne usluge tijekom trajanja ispitivanja. To znači da se manje podataka izdvaja iz sadržaja poruka i manje je zapisa o korištenju Ispitne usluge potrebno slati u središnji spremnik. Budući da je veličina zapisa o korištenju Ispitne usluge jednaka 1,25 kilookteta [KB], a i promatrana je srednja vrijednost mrežnog prometa od računala, slanje zapisa o korištenju Ispitne usluge ne utječe u velikoj mjeri na mrežni promet od računala.



Slika 7.9: Ukupan mrežni promet računala u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

Slika 7.9 prikazuje ukupni mrežni promet računala domaćina 1 u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*. Za vrijednosti *T* manje od 5 sekundi [s] ukupni mrežni promet protokola za potiskivanje veći je od ukupnog mrežnog promete protokola za istiskivanje. Za vrijednosti *T* veće i jednake 5 sekundi

[s] ukupan mrežni promet oba protokola za razmjenu podataka gotovo je identičan. Budući da je ukupan mrežni promet jednak zbroju mrežnog prometa od i prema računalu analiza primijenjena na prošla dva grafa može se primijeniti na ovom grafu. Vrijednost vremenskog razmaka između dva uzastopna poziva usluge obrnuto je proporcionalna broju ostvarenih poziva Ispitne usluge. Manje vrijednosti vremenskog razmaka između dva uzastopna poziva usluge znače više poziva Ispitne usluge. Ako postoji više poziva Ispitne usluge onda postoji i više zapisa o korištenju Ispitne usluge iz čega proizlazi razlika protokola u ukupnom mrežnom prometu za manje vrijednosti vremenskog razmaka između dva uzastopna poziva usluge. Veće vrijednosti vremenskog razmaka između dva uzastopna poziva usluge znače manje poziva Ispitne usluge. Ako postoji manje poziva Ispitne usluge onda postoji i manje zapisa o korištenju Ispitne usluge. Uz manji broj zapisa o korištenju Ispitne usluge i uz veličinu zapisa o korištenju Ispitne usluge od 1.25 kilookteta [KB] proizlazi da je razlika protokola u ukupnom mrežnom prometu minimalna za veće vrijednosti vremenskog razmaka između dva uzastopna poziva usluge.



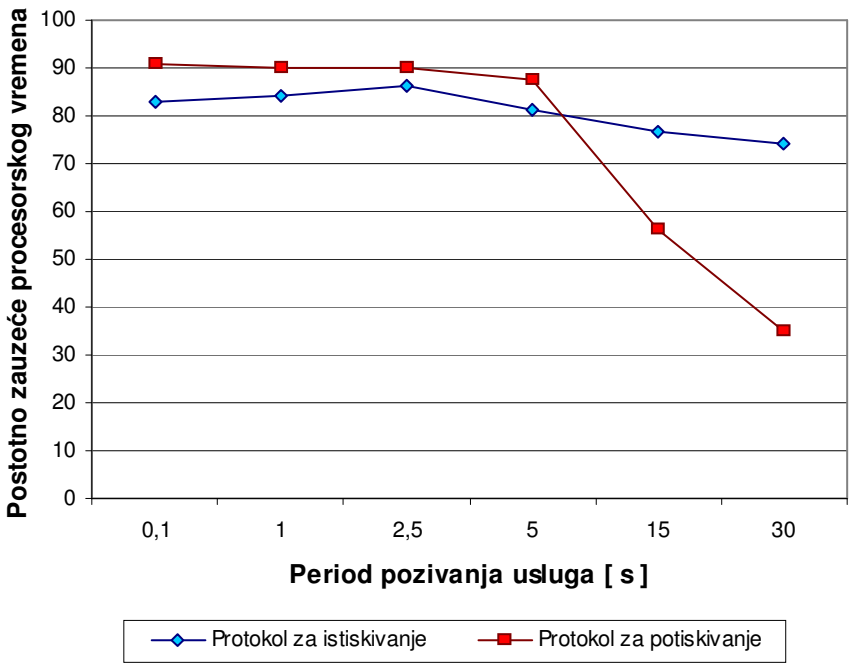
Slika 7.10: Veličina izdvojenih zapisa Ispitne usluge u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

Slika 7.10 prikazuje ukupnu veličinu izdvojenih zapisa o korištenju Ispitne usluge tijekom trajanja ispitivanja. U slučaju protokola za istiskivanje mjerila se veličina datoteke lokalnog spremnika na računalu domaćinu 5, a u slučaju protokola

za potiskivanje mjerena je veličina datoteke središnjeg spremnika na računalu domaćinu 1. Za sve vrijednosti vremenskog razmaka između dva uzastopna poziva usluge ukupna veličina zapisa o korištenju kod jednog i drugog protokola za razmjenu podataka gotovo je identična. Iz toga se zaključuje da je uporabom oba protokola za vrijeme trajanja ispitivanja ostvaren gotovo jednak broj poziva Ispitne usluge.

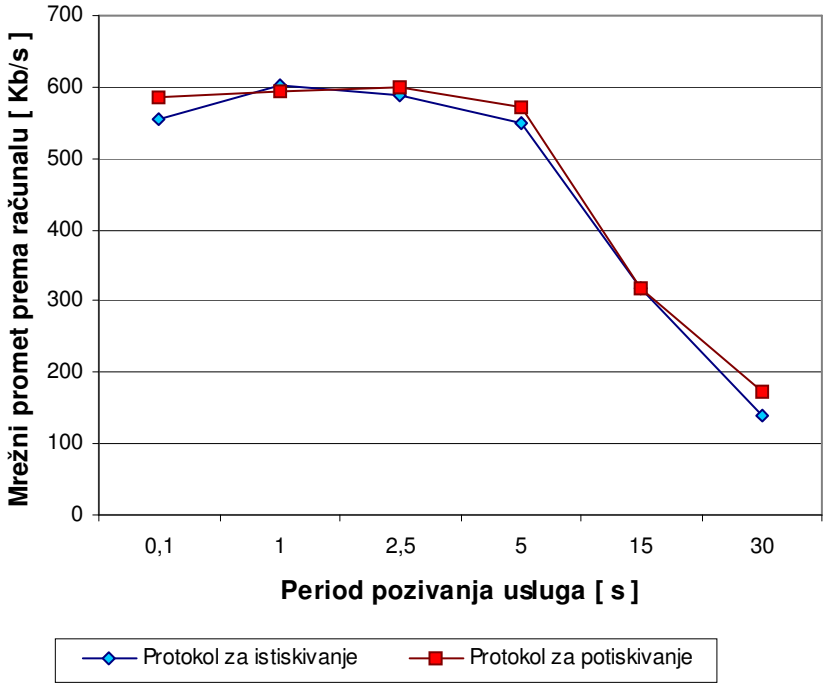
Iz ove serije mjerenja može se zaključiti da primjena oba protokola za razmjenu podataka Sustava za nadgledanje rada korisnika i usluga ostvaruje gotovo jednak broj poziva Ispitne usluge. Protokol za potiskivanje ima manje postotno procesorsko zauzeće u odnosu na protokol za istiskivanje. Razlika u postotnom zauzeću povećava se kako se povećava vrijednost vremenskog razmaka između dva uzastopna poziva usluge. Razlika u mrežnom prometu očituje se pri vrijednostima vremenskog razmaka između dva uzastopna poziva usluge manjima od 15 sekundi [s], nakon kojih je sve manja. Stoga se može reći da u okolinama gdje je cijena slanja zapisa o korištenju usluga prihvatljiva, veličina središnjeg spremnika dovoljno velika i veličina zapisa o korištenju usluga mala isplativije koristiti protokol za potiskivanje. U nastavku je prikazana druga podskupina rezultata s ulaznim argumentom veličine 1000 kilookteta [KB].

Slika 7.11 prikazuje postotno procesorsko zauzeće na računalu domaćinu 1 u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*. Pri vrijednostima *T* do 5 sekundi [s] protokol za potiskivanje ima veće postotno zauzeće procesora u odnosu na protokol za istiskivanje. Kako vrijednosti *T* rastu tako postotno zauzeće kod protokola za potiskivanje opada strmije u odnosu na protokol za istiskivanje. Za vrijednosti *T* veće od 15 sekundi [s] razlika u postotnom zauzeću je 20% i ona raste kako raste vrijednost *T*. Uz korištenje već prikazanog obrasca za nadgledanje Ispitne usluge Podsustav za lokalno nadgledanje pri svakom pozivu Ispitne usluge, uz veličinu ulaznog argumenta od 1000 kilookteta [KB], stvara dva zapisa veličine 125 kilookteta [KB]. Na osnovi grafa može se zaključiti da operacija zapisivanja u lokalni spremnik, kada je veličina zapisa o korištenju usluge relativno velika i vrijednosti *T* manje od 5 sekundi [s], troši manje procesorskog vremena u odnosu na operaciju slanja zapisa o korištenju usluge putem računalne mreže. Ali kako vrijednost *T* raste operacija zapisivanja u lokalni spremnik postaje skuplja



Slika 7.11: Procesorsko zauzeće u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

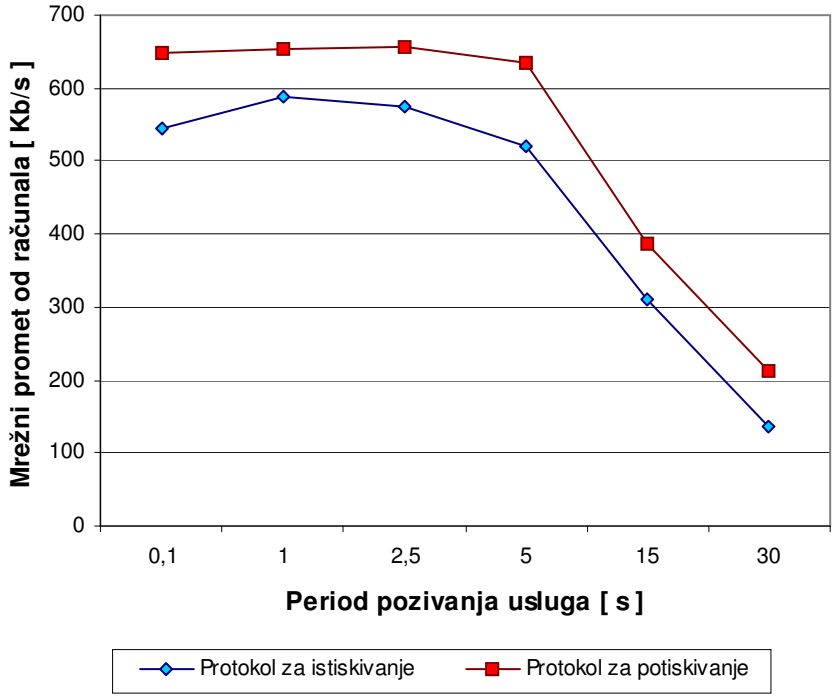
operacija od operacije slanja zapisa o korištenju usluge računalne mreže. U ovom slučaju skuplja operacija znači da operacija ima veće postotno zauzeće procesora.



Slika 7.12: Mrežni promet prema računalu u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

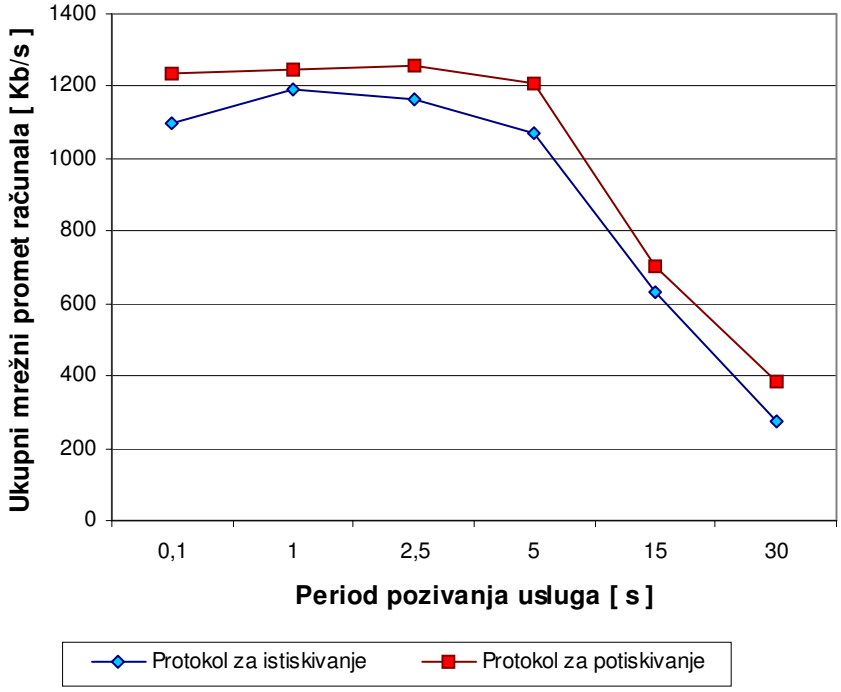
Slika 7.12 prikazuje mrežni promet prema računalu u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T* na računalu domaćinu 1. Za sve vrijednosti *T* mrežni promet prema računalu oba protokola gotovo je jednakih iznosa što je u skladu s očekivanjima.

Slika 7.13 prikazuje mrežni promet od računala u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T* na računalu domaćinu 1. Za sve vrijednosti *T* mrežni promet od računala protokola za potiskivanje veći je od mrežnog prometa od računala protokola za istiskivanje što je posljedica prijenosa zapisa o korištenju Ispitne usluge u središnji spremnik. Za vrijednosti *T* veće od 5 sekundi razlika u iznosu mrežnog prometa od računala ostaje konstantna.



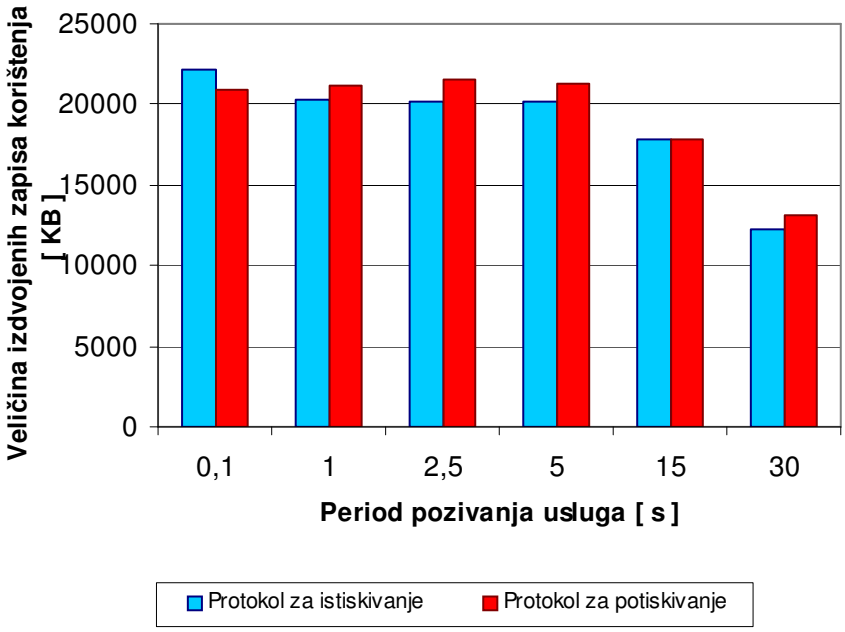
Slika 7.13: Mrežni promet od računalu u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

Slika 7.14 prikazuje ukupan mrežni promet računala u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T* na računalu domaćinu 1. Ukupan mrežni promet je suma mrežnog prometa od i prema računalu. Mrežni promet od računala oba protokola za razmjenu podataka gotovo je jednak, a mrežni promet od računala protokola za potiskivanje veći od mrežnog prometa od računala protokola za potiskivanje za sve vrijednosti *T*. Stoga je za očekivati da je ukupni



Slika 7.14: Ukupan mrežni promet računala u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge *T*

mrežni promet protokola za potiskivanje veći od ukupnog mrežnog prometa protokola za istiskivanje za sve vrijednosti *T* što je i vidljivo na grafu.



Slika 7.15: Veličina izdvojenih zapisa Ispitne usluge u ovisnosti o vremenskom razmaku između dva uzastopna poziva usluge

Slika 7.15 prikazuje veličinu izdvojenih zapisa Ispitne usluge. U slučaju protokola za istiskivanje mjerila se veličina datoteke lokalnog spremnika na računalu domaćinu 5, a u slučaju protokola za potiskivanje mjerila se veličina datoteke središnjeg spremnika na računalu domaćinu 1. Iako su razlike u ukupnoj veličini zapisa o korištenju Ispitne usluge izraženije nego u prvoj seriji mjerenja, mora se uzeti u obzir da je veličina zapisa o korištenju Ispitne usluge u ovim mjerenjima jednaka 125 kilookteta [KB]. Najveća razlika u veličini zapisa o korištenju Ispitne usluge je izmjerena pri vrijednosti *T* 2,5 sekunde [s] i iznosi 1 321 kilooktet [KB], što predstavlja 6,5% veličine zapisa o korištenju Ispitne usluge protokola za istiskivanje. Može se reći da 1 321 kilooktet [KB] zapisa o korištenju Ispitne usluge predstavlja 10 poziva Ispitne usluge. Ukupna veličina izdvojenih zapisa o korištenju Ispitne usluge protokola za istiskivanje iznosi 20 189 kilooktet [KB] ili otprilike 161 poziv Ispitne usluge, a veličina zapisa o korištenju Ispitne usluge protokola za potiskivanje iznosi 21 510 kilookteta [KB] ili otprilike 172 poziva. Stoga se može zaključiti da je razlika u broju poziva manja od 6.5%.

Iz ove serije mjerenja može se zaključiti da pri većim količinama zapisa o korištenje usluga dolazi do razlike opterećenja računalne mreže u radu protokola za razmjenu podataka. Pri manjim vrijednostima vremenskog razmaka između dva uzastopna poziva usluge protokol za potiskivanje u odnosu na protokol za istiskivanje ima veće procesorsko postotno opterećenje, ali se pri većim vrijednostima situacija mijenja.

Mjerenja pokazuju da izbor protokola za razmjenu podataka Sustava za nadgledanje rada korisnika i usluga ovisi o velikom broj faktora. Broj usluga čiji se rad prati i korisnika koji pozivaju praćene usluge znatno će utjecati na opterećenje računalne mreže. Opterećenje pozivanih metoda praćenih usluga i veličina podataka koji se izdvajaju iz poruka također znatno utječe na opterećenje računalne mreže. Vrijeme između dva uzastopna poziva praćene usluge na računalu domaćinu znatno utječe na procesorsko opterećenje. Veličina lokalnih spremnika i veličina središnjeg spremnika znatno utječe na izbor protokola za razmjenu podataka. Također je vrijeme trenutka kad su raspoloživi svi prikupljeni zapisi o korištenju usluge znatan faktor u izboru protokola. Stoga je za svaku radnu okolinu potrebno prikupiti sve navedene podatke, postaviti kriterije rada, provesti ispitivanje rada Sustava za

nadgledanje rada korisnika i usluga i donijeti konačnu odluku koja je u skladu s dobivenim rezultatima mjerenja.

8 Zaključak

Sustavi za ispitivanje radnih značajki važni su budući da olakšavaju postupak upravljanja i ispitivanja raspodijeljenih sustava. Primjenom sustava za ispitivanje radnih značajki prikupljaju se informacije o računalnoj i mrežnoj infrastrukturi te primjenskim programima. Na temelju prikupljenih informacija upravitelji sustava otkrivaju kvarove i nedostatke u radu raspodijeljenog sustava.

U diplomskom radu opisan je model i programsko ostvarenje sustava za ispitivanje radnih značajki u okolinama zasnovanim na uslugama. Opisana je programska arhitektura ostvarenog sustava i prikazan je postupak upravljanja radom sustava. Korisnik upravlja radom sustava zadavanjem skupa radnih postavki na temelju kojih se provode ispitivanja. Tijekom provođenja ispitivanja, Sustav za ispitivanje radnih značajki stvara opterećenje za ispitivani sustav i mjeri njegove radne značajke. Na kraju ispitivanja Sustav za ispitivanje radnih značajki usluga prikuplja sve radne značajke i stvara izlaznu datoteku radnih značajki. Primjenom ostvarenog sustava za ispitivanje izvršeno je ispitivanje radnih značajki Sustava za nadgledanje rada korisnika i usluga. Na temelju prikupljenih rezultata ispitivanja provedena je analiza radnih značajki Sustava za nadgledanje rada korisnika i usluga u različitim režimima rada. Dobiveni rezultati omogućili su definiranje tipičnih zahtjeva u pogledu količine računalnih sredstava potrebnih za rad Sustava za nadgledanje rada korisnika i usluga u različitim režimima rada.

Cilj budućeg rada je omogućiti prikupljanje i prikazivanje radnih značajki upravitelju sustava u stvarnom vremenu te prikazivanje radnih značajki primjenom odgovarajućih grafičkih sučelja. Nadalje, ciljevi budućeg rada bit će i proširivanje skupa računalnih sredstava za koje je moguće ostvariti nadgledanje rada izgradnjom dodatnih vrsta ispitnih sondi.

9 Literatura

[1] N. Walsh: **"A Technical Introduction to XML",** October 1998., (http://www.xml.com/pub/a/98/10/guide0.html)

[2] D. Wellman: **"An Introduction to XML Schemas"**, November 2004., (http://www.devarticles.com/c/a/XML/An-Introduction-to-XML-Schemas/1/)

[3] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, et al.: **"Simple Object Access Protocol (SOAP) 1.1"**, May 2000., (http://www.w3.org/TR/2000/NOTE-SOAP-20000508/)

[4] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana: **"Web Services Description Language (WSDL) 1.1"**, March 2001., (http://www.w3.org/TR/wsdl)

[5] K. Scribner, M. C. Stiver: **"Applied SOAP: Implementing .NET XML Web Services"**, Sams Publishing, October 2001.

[6] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, D. F. Ferguson: **"Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More"**, Prentice Hall PTR, 2005.

[7] *RSP product information manual*, (http://www.dracoware.com/RSP-Product-Brochure.pdf)

[8] *Heroix Longitude Brochure*, (http://www.heroix.com/downloads/pdf/Longitude_Brochure.pdf)

[9] *eG Enterprise Suite*, (http://www.eginnovations.com/web/)

[10] *eG Features Summary*, (http://www.eginnovations.com/web/featuresummary.htm)

[11] D. Škvorc: **"Prividna mreža računalnih sustava zasnovanih na uslugama"**, Magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[12] M. Podravec: **"Otkrivanje i postavljanje usluga u sustavima zasnovanima na uslugama"**, Magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[13] A. Milanović: **"Programski model zasnovan na uslugama"**, Doktorska disertacija, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.

[14] D. Skrobo: **"Raspodijeljeno usporedno interpretiranje programa u arhitekturama zasnovanim na uslugama"**, Magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[15] M. Popović: **"Nadziranje pristupa računalnim sustavima zasnovanim na uslugama"**, Magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[16] K. Vladimir: **"Sustav za nadgledanje rada korisnika i usluga u razvojnoj okolini PIE"**, Diplomski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2007.

[17] *.NET Framework Developer Center*, (http://msdn2.microsoft.com/en-us/netframework/default.aspx)

[18] *The Official Microsoft ASP .NET 2.0 Site*, (http://www.asp.net/)

[19] *Internet Information Services*, (http://www.microsoft.com/WindowsServer2003/iis/default.mspx)

[20] *Web Services Enhancements (WSE)*, (http://msdn2.microsoft.com/en-us/webservices/Aa740663.aspx)