

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Dejan Škvorc

**PRIVIDNA MREŽA RAČUNALNIH SUSTAVA
ZASNOVANIH NA USLUGAMA**

MAGISTARSKI RAD

Zagreb, 2006.

Magistarski rad je izrađen u Zavodu za elektroniku,
mikroelektroniku, računalne i inteligentne sustave
Fakulteta elektrotehnike i računarstva
Sveučilišta u Zagrebu

Mentor: Prof.dr.sc. Siniša Srbljić

Magistarski rad ima 197 stranica.

Rad br.: _____

Povjerenstvo za ocjenu u sastavu:

1. Akademik prof.dr.sc. Leo Budin – predsjednik
2. Prof.dr.sc. Siniša Srbljić – mentor
3. Doc.dr.sc. Saša Dešić – Ericsson Nikola Tesla, Zagreb

Povjerenstvo za obranu u sastavu:

1. Akademik prof.dr.sc. Leo Budin – predsjednik
2. Prof.dr.sc. Siniša Srbljić – mentor
3. Doc.dr.sc. Saša Dešić – Ericsson Nikola Tesla, Zagreb

Datum obrane: 25. siječnja 2006.

Zahvala

Od ljudi kojima sam bio okružen od najranijih dana, koji su mi uvek bili spremni pružiti podršku i koji su me učili životu, naučio sam važnu pouku. Najvažnije mudrosti nisu zapisane u knjigama, već se rađaju iz svakodnevnih životnih situacija. Život je vrlo dugačka i teška utrka u kojoj čovjek prije svega mora naučiti pobjeđivati sam sebe i svoje vlastite unutarnje strahove. Shvatio sam da mi svaki novi uspjeh mora biti pouka da u budućnosti ne ponovim iste pogreške, a svaki novi uspjeh poticaj za dostizanje još neostvarenih ciljeva. Moja je želja, dužnost i obveza spomenuti one koji su mi u tome pomagali.

Zahvaljujem prof. dr. sc. Siniši Srbljiću na mentorstvu i stručnom vodstvu tijekom dodiplomskog i poslijediplomskog studija, te tijekom izrade magistarskog rada. Zahvaljujem mu na stvaranju poticajne radne sredine i savjetima koji su u mnogome pridonijeli mojem stručnom i osobnom razvoju.

Zahvalu upućujem svim profesorima i znanstvenicima čijim je zalaganjem pokrenut tehnologiski projekt CroGrid, čime mi je omogućeno zapošljavanje i istraživački rad na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Tvrzci Ericsson Nikola Tesla d.d. iz Zagreba iskazujem zahvalnost na uspješnoj suradnji i finansijskoj potpori istraživačkom radu u okviru kojega je izrađen magistarski rad.

Zahvaljujem svim članovima Zavoda za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva na stvaranju poticajne radne sredine, a naročito suradnicima na projektu CroGrid dr.sc. Andri Milanoviću, mr.sc. Ivanu Skuliberu, mr.sc. Ivanu Bencu, Danielu Skrobi, dipl.ing., Matiji Podravcu, dipl.ing., Ivanu Gavranu, dipl.ing. i Miroslavu Popoviću, dipl.ing.

Posebna zahvala suradnicima mr.sc. Ivanu Skuliberu i Danielu Skrobi, dipl.ing. te sestrični Aniti Škvorc, prof. na čitanju teksta magistarskog rada i brojnim korisnim savjetima i preporukama za poboljšanje njegove kakvoće.

Najveću zahvalnost želim izraziti djevojci Mariji na podršci, strpljenju i razumijevanju zbog mojih čestih odsutnosti duhom i tijelom za vrijeme istraživačkog rada i pisanja teksta magistarskog rada. Hvala joj na riječima ohrabrenja koje su mi nebrojeno puta bile potrebne za ustrajnost u postizanju zahtjevnih ciljeva. Njezina neizmjerna spremnost na odricanje postala je dijelom ovog rada.

Naposljeku, zahvaljujem roditeljima, bratu i prijateljima na moralnoj podršci i razumijevanju.

Dejan Škvorc

Sadržaj

Sadržaj.....	i
1 Uvod	1
2 Računarstvo zasnovano na uslugama	4
2.1 Arhitektura zasnovana na uslugama.....	6
2.1.1 Programske usluge.....	6
2.1.2 Slabo povezivanje.....	8
2.1.3 Komunikacija zasnovana na razmjeni poruka	10
2.1.4 Osnovne komponente arhitekture zasnovane na uslugama.....	10
2.1.5 Povezivanje usluga	12
2.1.6 Tehnologije i standardi za izgradnju sustava zasnovanih na uslugama	13
2.2 Usporedba oblikovanja zasnovanog na uslugama s oblikovanjem zasnovanim na objektima i komponentama	15
3 Prividne logičke mreže	20
3.1 Komunikacijski podsustav prividne logičke mreže.....	22
3.1.1 Upravljanje oblikom i ustrojem prividne mreže	24
3.1.2 Naslovljavanje logičkih čvorova	25
3.1.3 Pretraživanje i usmjeravanje sadržaja.....	27
3.1.4 Sigurnost prijenosa podataka	32
3.2 Mreže ravnopravnih sudionika.....	37
3.2.1 Arhitektura mreže ravnopravnih sudionika	38
3.2.2 Programski međusloj mreža ravnopravnih sudionika	42
3.3 Zgodom oblikovane mreže pokretnih sudionika	48
3.3.1 Upravljanje pokretljivošću sudionika	52
3.3.2 Usmjeravanje sadržaja	54
4 Programski međusloj zasnovan na razmjeni poruka	64
4.1 Modeli međudjelovanja u komunikacijskim sustavima	64
4.1.1 Sinkrona komunikacija	65
4.1.2 Asinkrona komunikacija	67
4.1.3 Analiza primjenjivosti modela međudjelovanja u komunikacijskim sustavima....	70
4.2 Metode razmjene poruka	75
4.2.1 Prozivanje	76
4.2.2 Povratni poziv.....	78
4.2.3 Objava/pretplata.....	79
4.2.4 Analiza primjenjivosti metoda razmjene poruka	81

5 Prividna raspodijeljena računalna okolina.....	83
5.1 Programski model zasnovan na uslugama.....	84
5.1.1 Faza oblikovanja raspodijeljenog programskog sustava.....	86
5.1.2 Faza programskog ostvarenja raspodijeljenog programskog sustava	90
5.2 Arhitektura prividne raspodijeljene računalne okoline	91
6 Prividna mreža računalnih sustava zasnovanih na uslugama	96
6.1 Smjernice za oblikovanje i izgradnju prividne mreže	96
6.2 Mreža logičkih čvorova.....	98
6.3 Naslovljavanje usluga u komunikacijskom prostoru prividne mreže	100
6.4 Logički komunikacijski protokol prividne mreže	102
6.5 Prenosivost raspodijeljenih programskih sustava.....	104
6.6 Otpornost raspodijeljenih programskih sustava na kvarove mrežnih čvorova.....	106
6.7 Potpora prostornoj lokalnosti usluga	108
6.8 Usmjeravanje poruka.....	109
6.9 Sigurnost komunikacije.....	112
7 Arhitektura prividne mreže računalnih sustava zasnovanih na uslugama	115
7.1 Usmjernik poruka	118
7.1.1 Struktura poruke za ostvarivanje komunikacije logičkih čvorova	120
7.1.2 Arhitektura usmjernika poruka	122
7.1.3 Isporuka poruke lokalnim programskim uslugama.....	123
7.1.4 Isporuka poruke udaljenim logičkim čvorovima	126
7.2 Imenik lokalnih usluga	129
7.2.1 Tablica pristupnih točaka.....	129
7.2.2 Tablica prava pristupa.....	130
7.2.3 Arhitektura imenika lokalnih usluga.....	132
7.2.4 Prijava lokalnih programskih usluga u sustav prividne mreže	134
7.3 Spremnik zastupnika lokalnih usluga.....	136
7.4 Generator zastupnika lokalnih usluga	138
7.5 Upravitelj relacijom preslikavanja i ustrojem prividne mreže	140
7.5.1 Tablica relacije preslikavanja	140
7.5.2 Tablica usmjeravanja	141
7.5.3 Arhitektura upravitelja relacijom preslikavanja i ustrojem prividne mreže	144
7.6 Zastupnik usluge prividne mreže udaljenog logičkog čvora	145
7.7 Upravitelj članstvom prividne mreže	146
7.8 Jedinica za sigurnosnu zaštitu poruka	149
8 Programsко ostvarenje prividne mreže računalnih sustava zasnovanih na uslugama ...	153

8.1 Usmjernik poruka	154
8.2 Imenik lokalnih usluga, Upravitelj relacijom preslikavanja i ustrojem prividne mreže i Spremnik zastupnika lokalnih usluga	159
8.3 Generator zastupnika lokalnih usluga	162
8.4 Zastupnik usluge prividne mreže udaljenog logičkog čvora	165
8.5 Upravitelj članstvom prividne mreže	167
8.6 Jedinica za sigurnosnu zaštitu poruka	168
8.7 Prevođenje i postavljanje programske usluge prividne mreže	171
9 Zaključak	179
10 Literatura	182
11 Životopis	193
12 Sažetak	195
13 Summary	196
14 Ključne riječi	197

1 Uvod

Kraj dvadesetog i početak dvadeset i prvog stoljeća u analima računarskih i informacijskih znanosti ostat će zapamćeni po prodiranju globalne računalne mreže Internet u sve pore suvremenog društva. Iako prvo bitno razvijena kao obavještajna mreža američke vojske otporna na sve vrste vojnog djelovanja, današnju prihvaćenost u civilnom društvu mreža Internet ponajprije duguje znanstvenoj zajednici. Nakon nekoliko desetljeća razvoja i usavršavanja u okviru znanstvenih istraživanja, mreža Internet dosegnula je zadovoljavajuću razinu pouzdanosti i prihvaćenosti za primjenu u poslovnim, zabavnim, obrazovnim, upravnim i sličnim područjima ljudske djelatnosti.

Široka prihvaćenost globalne mreže Internet posljedica je dobro definiranih, sveobuhvatnih i standardima usvojenih tehnologija za prijenos podataka zasnovanih na *IP* protokolu. Primjena *IP* protokola, kao osnovnog komunikacijskog protokola globalne mreže Internet, omogućila je stvaranje višenamjenske komunikacijske infrastrukture, neovisne o područjima primjene. Prilagodba osnovne komunikacijske infrastrukture posebnostima pojedinih područja primjene ostvaruje se razvojem primjenskih komunikacijskih protokola iznad osnovnog *IP* protokola. Mogućnost nadogradnje *IP* protokola primjenskim protokolima omogućila je razdvajanje funkcionalnosti komunikacijske mrežne infrastrukture od uslužnih funkcionalnosti područja primjene. Razdvajanjem komunikacijskih i uslužnih funkcionalnosti pojednostavljuje se razvoj raznovrsnih i raznorodnih mrežnih usluga u zajedničkom komunikacijskom prostoru mreže Internet. Upravo je mogućnost istovremenog postojanja i djelovanja raznovrsnih, raznorodnih i međusobno neuskladivih primjenskih programskih tehnologija potaknula nagli razvoj raspodijeljenih programskih sustava zasnovanih na mreži Internet te njezino prihvaćanje kao okosnice globalno povezanog informacijskog društva.

Težnje suvremenog informacijskog društva ubrzo su, međutim, nadiše okvire globalne povezivosti na komunikacijskoj razini i okrenule se povezivanju programskih sustava različitih područja primjene. Raznovrsnost, raznorodnost i neuskladivost programskih tehnologija primjenske razine razlozi su zbog kojih većina današnjih međudjelovanja različitih aplikacija zahtjeva posredovanje čovjeka. Najnoviji pravci znanstvene zajednice u području istraživanja i razvoja programskih tehnologija za primjenu u globalnoj mreži Internet usmjereni su mogućnostima povezivanja raznovrsnih i raznorodnih aplikacija i njihovog samostalnog međudjelovanja i održavanja. Povezivost raznovrsnih i raznorodnih programskih sustava osnovna su načela računarstva zasnovanog na uslugama. Računalni sustavi zasnovani na uslugama grade se objedinjavanjem programskih usluga različitih

proizvođača, namijenjenih različitim područjima primjene te ponuđenih na korištenje putem mreže Internet od strane različitih pružatelja usluga. Raznorodnost programskih tehnologija korištenih pri izgradnji programskih usluga prevladana je usvajanjem skupa standardnih tehnologija za objedinjavanje programske potpore, slično kao što je standardom usvojeni *IP* protokol postao okosnicom globalne komunikacijske infrastrukture.

Poželjna svojstva raspodijeljenih aplikacija izgrađenih povezivanjem usluga razasutih diljem mreže Internet za primjenu u poslovanju, zabavi, obrazovanju, zdravstvu, upravi i ostalim područjima svakodnevnog života su učinkovitost, otpornost na kvarove, sigurnost te mogućnost prenosivosti između različitih radnih okolina. Svojstva učinkovitosti, otpornosti na pogreške, sigurnosti i prenosivosti raspodijeljenih aplikacija zasnovanih na uslugama moguće je postići izgradnjom aplikacija u komunikacijskom prostoru prividne mreže. Prividna mreža je raspodijeljeni programski sustav koji pruža komunikacijsku infrastrukturu za učinkovito i sigurno povezivanje programskih usluga u prenosive i na kvarove otporne raspodijeljene sustave zasnovane na uslugama. Prividna mreža računalnih sustava zasnovanih na uslugama, oblikovana, ostvarena i opisana u okviru ovog magistarskog rada, dio je prividne raspodijeljene računalne okoline. Prividna raspodijeljena računalna okolina je raspodijeljeni programski sustav za potporu oblikovanju, razvoju, postavljanju i izvođenju raspodijeljenih aplikacija zasnovanih na uslugama. Razvijena je i programski ostvarena u suradnji Zavoda za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu i Instituta za telekomunikacije tvrtke Ericsson Nikola Tesla d.d. iz Zagreba. Rezultati zajedničkog istraživačkog rada iskorišteni su kao polazna točka za pokretanje tehnologiskog projekta *CroGrid* pod pokroviteljstvom Ministarstva znanosti, obrazovanja i športa Republike Hrvatske.

U ovom radu predložena je arhitektura te je opisano programsko ostvarenje prividne mreže računalnih sustava zasnovanih na uslugama. Osnovna namjena sustava prividne mreže je potpora za izgradnju i održavanje raspodijeljenih aplikacija zasnovanih na uslugama sa svojstvima prenosivosti, učinkovitosti, sigurnosti i otpornosti na kvarove fizičke mrežne infrastrukture. Izgrađeni sustav prividne mreže omogućava uspostavljanje logičkog komunikacijskog prostora za raspodjelu programskih usluga i njihovu međusobnu komunikaciju. Sustavom naslovljavanja usluga nezavisnim o sustavu adresiranja čvorova fizičke mreže postignuta je neovisnost izgrađenih aplikacija o načinu adresiranja i broju raspoloživih mrežnih čvorova u fizičkim radnim okolinama. Mogućnost dinamičke prilagodbe sustava naslovljavanja prividne mreže uvjetima fizičke radne okoline omogućuje izgradnju mehanizama otpornosti raspodijeljenih aplikacija zasnovanih na uslugama na kvarove mrežnih čvorova po kojima su raspodijeljene funkcionalnosti u obliku programskih

usluga. Sustavom logičkog usmjeravanja poruka odvojenim od usmjeravanja poruka na razini fizičke mreže podignuta je razina privatnosti korisnika i pružatelja programskih usluga.

Magistarski rad podijeljen je u dvije cjeline. Prvom cjelinom obuhvaćen je pregled dosadašnjih rezultata istraživačkog rada u području računarstva zasnovanog na uslugama, prividnih mreža i komunikacijskih modela u raspodijeljenim računalnim sustavima. Tom su cjelinom obuhvaćena prva četiri poglavlja magistarskog rada. Osnovne smjernice računarstva zasnovanog na uslugama te razvoj raspodijeljenih aplikacija zasnovanih na uslugama opisani su u drugom poglavlju. Treće poglavlje daje pregled najvažnijih svojstava prividnih logičkih mreža i mogućnosti njihove primjene za izgradnju raspodijeljenih aplikacija sa svojstvima neograničenog razmjernog rasta. U četvrtom poglavlju razmatraju se modeli komunikacijskog međudjelovanja među elementima raspodijeljenih računalnih sustava. Posebna pažnja posvećena je modelu komunikacije zasnovanom na razmjeni poruka kao komunikacijskoj paradigmi za razvoj raspodijeljenih aplikacija sa svojstvima neograničenog razmjernog rasta i otpornosti na pogreške.

Drugom cjelinom prikazani su rezultati teorijskog i praktičnog istraživačkog rada u području dodijeljene teme magistarskog rada. Ovom je cjelinom obuhvaćeno pet završnih poglavlja. U petom poglavlju predstavljena je prividna raspodijeljena računalna okolina. Jedna od funkcionalnosti prividne raspodijeljene računalne okoline je mogućnost oblikovanja prividne mreže za raspodjelu programskih funkcionalnosti u obliku programskih usluga. Svojstva prividne mreže računalnih sustava zasnovanih na uslugama i smjernice za njezino programsko ostvarenje izneseni su u šestom poglavlju. Sedmo poglavlje donosi opis arhitekture programskog sustava prividne mreže računalnih sustava zasnovanih na uslugama, dok je u osmom poglavlju opisano njegovo programsko ostvarenje. Zaključna riječ i osvrt na ostvareni programski sustav prividne mreže izneseni su u devetom poglavlju.

2 Računarstvo zasnovano na uslugama

Razvoj složenih programskih sustava prolazio je tijekom vremena kroz nekoliko faza. Svaka je faza uvodila određena poboljšanja koja su postupak razvoja programskih sustava činila bržim, lakšim i jednostavnijim. Već u vrlo ranoj fazi razvoja uočeno je znatno produljenje vremena razvoja i potreba za povećanim naporima graditelja uzrokovana porastom složenosti programske potpore. Kako bi se prevladale poteškoće uzrokovane porastom složenosti programske potpore, rodila se zamisao o organizaciji programskog kôda u module te višestrukom korištenju jednom napisanih programskih modula. Višestruka iskoristivost (engl. *reusability*) programskih modula bila je krajnji doseg paradigme proceduralnog programiranja. Veliki nedostatak iskorištavanja programskog kôda u obliku programskih modula dolazio je do izražaja za vrijeme održavanja programske potpore. Ako se javila potreba za ispravljanjem programske pogreške u jednom od modula, promjene su se provodile na svim mjestima na kojima se koristio programski kôd tog modula.

Paradigmu proceduralnog programiranja postupno je zamijenjena objektno orijentirana paradigma (engl. *object-oriented programming*) [1]. Objektno orijentirana paradigma uvodi pojmove razreda i primjeraka razreda, odnosno objekata. Korištenjem objektno-orijentiranog pristupa programiranju, programski sustavi se izgraduju kao skupina razreda. Razredi su apstraktni tipovi podataka koji predstavljaju modele objekata iz stvarnog svijeta. Svaki razred ima definiranu programsku funkcionalnost koju ostvaruje, strukture podataka nad kojima se ta funkcionalnost obavlja i programsko sučelje putem kojeg drugi razredi s njim ostvaruju vezu. Višestruko korištenje jednom napisanog programskog kôda pojedinih razreda omogućena je mehanizmima stvaranja primjeraka razreda, odnosno objekata, i nasljeđivanja funkcionalnosti iz prethodno razvijenih razreda. Prednosti objektno orijentiranog pristupa programiranju te načela stvaranja primjeraka razreda i nasljeđivanja razreda vidljive su tijekom izgradnje i za vrijeme održavanja programske potpore. Tijekom izgradnje programske potpore, razredi s novim programskim funkcionalnostima ostvaruju se nasljeđivanjem funkcionalnosti od prethodno ostvarenih razreda. Pri uočenim programskim pogreškama za vrijeme održavanja izgrađenog sustava, ispravke se trebaju provesti nad programskim kôdom razreda, a na svim mjestima gdje se koriste objekti tog razreda promjene su automatski vidljive.

Kako bi se dodatno olakšao i ubrzao razvoj složenih programskih sustava, osmišljena je paradigma oblikovanja koja se zasniva na višestrukoj iskoristivosti programskih funkcionalnosti u više nezavisnih sustava, a ne samo iskoristivosti programskog kôda unutar jednog sustava. Višestruka iskoristivost programskih funkcionalnosti omogućena je

primjenom programske paradigme zasnovane na komponentama (engl. *component-based development*) [2, 3]. Programska komponenta je dio programskog sustava koji obavlja dobro definiran i nezavisni skup funkcionalnosti. Funkcionalnosti programske komponente dostupne su putem programskog sučelja komponente. Korištenjem oblikovanja zasnovanog na komponentama razvoj programskih sustava znatno je ubrzan jer se novi programski sustavi grade iskorištavanjem gotovih programskih komponenata drugih proizvođača dostupnih na tržištu. Za ostvarenje programskih komponenata uglavnom se koristi oblikovanje zasnovano na proceduralnoj ili objektno orijentiranoj paradigmi programiranja.

Primjenom programske paradigme zasnovane na komponentama omogućen je ubrzani razvoj programskih sustava. Programski sustavi grade se pronalaskom gotovih programskih komponenata na tržištu i njihovim sklapanjem u funkcionalne programske cjeline. Oblikovanje zasnovano na komponentama predstavlja zadovoljavajuće rješenje za izgradnju programskih sustava koji su u cjelini u nadležnosti jedne organizacije. Razvojem globalne mreže Internet i raspodijeljenog računarstva poslovne su organizacije svoje poslovanje sve više usmjeravale prema mrežnom poslovanju kako bi vlastite usluge mogle ponuditi na globalnom tržištu. Suvremeno shvaćanje poslovanja zahtjeva objedinjavanje (engl. *integration*) postojećih i novoizgrađenih programskih sustava različitih poslovnih organizacija. Programski sustavi zatvoreni u okvirima radne organizacije trebaju postati javno dostupni drugim organizacijama i korisnicima poslovnih usluga. Osnovni nedostatak paradigme oblikovanja programske potpore zasnovane na komponentama je problem raznorodnosti programskih tehnologija koje se koriste na globalnom tržištu i nemogućnost njihova međusobnog objedinjavanja.

Objedinjavanje raznorodnih programskih tehnologija osnovna je zamisao oblikovanja zasnovanog na uslugama (engl. *service-oriented design*) [4] koje čini okosnicu suvremenog pristupa izgradnji raspodijeljenih programskih sustava. Oblikovanje zasnovano na uslugama nastalo je proširenjem oblikovanja zasnovanog na komponentama, primjenjujući načela oblikovanja računalnih sustava za primjenu u tehnološki i organizacijski raznorodnim uvjetima globalne mreže Internet. Slično komponenti, usluga je osnovna programska jedinica od koje se grade raspodijeljeni programski sustavi zasnovani na uslugama. Ključne razlike između programske komponente i programske usluge su način upravljanja i vlasništvo nad sastavnim dijelovima složenog programskog sustava. Dok se programski sustav zasnovan na komponentama gradi kupovanjem komponenata raspoloživih na tržištu te njihovim samostalnim održavanjem, sustavi zasnovani na uslugama grade se korištenjem usluga ponuđenih na tržištu. Upravljanje i vlasništvo nad uslugom zadržava pružatelj usluge.

Na tržištu usluga programske usluge nisu ponuđene kao programske cjeline, već se naplaćuje njihovo korištenje.

Oblikovanje zasnovano na uslugama koristi se pri izgradnji složenih raspodijeljenih računalnih sustava sa svojstvima široke zemljopisne rasprostranjenosti, potrebe za dinamičkim povezivanjem pojedinih dijelova za vrijeme rada sustava te izvođenja u otvorenim i raznorodnim mrežnim okolinama. Naglasak programske paradigme zasnovane na uslugama stavljen je upravo na nezavisnost programskih usluga o programskim tehnologijama kojima su ostvarene njihove funkcionalnosti, kao i tehnološkim i organizacijskim uvjetima koji vladaju u okolini u kojoj se one izvode. Neovisnost o tehnološkim čimbenicima postiže se usvajanjem i primjenom standardnih tehnologija za izgradnju i objedinjavanje sustava zasnovanih na uslugama. Računarstvo zasnovano na uslugama (engl. *service-oriented computing*) [4, 10, 11] je programska paradigma koja izučava metodologiju oblikovanja, izgradnje, povezivanja i korištenja raspodijeljenih računalnih sustava čija je elementarna sastavna jedinica usluga. Računalni sustavi zasnovani na uslugama grade se na načelima arhitekture zasnovane na uslugama (engl. *service-oriented architecture*) [4, 10, 11, 12].

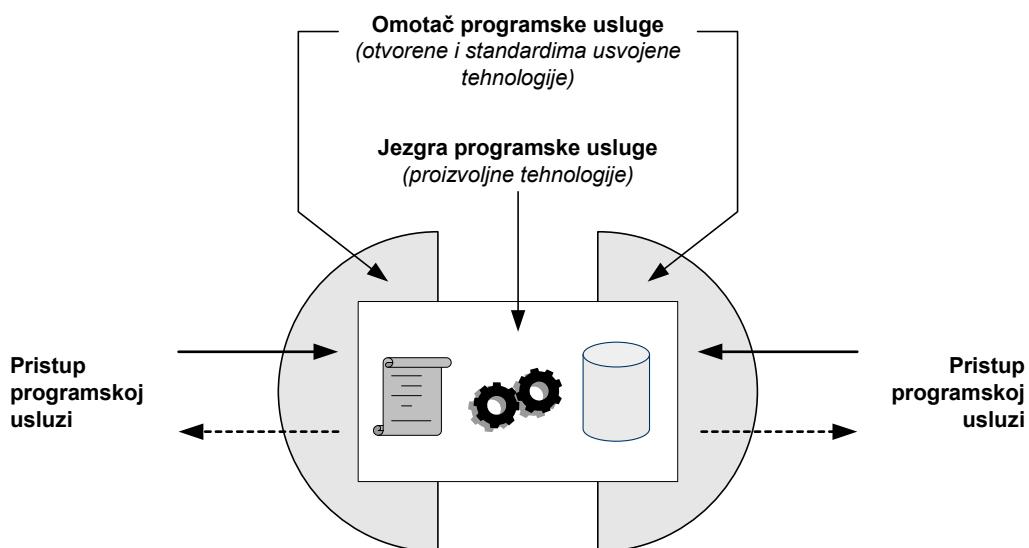
2.1 Arhitektura zasnovana na uslugama

Arhitektura zasnovana na uslugama definira skup pravila i smjernica za oblikovanje i izgradnju raspodijeljenih računalnih sustava u obliku skupa nezavisnih programskih usluga te njihovo povezivanje i objedinjavanje u cjelinu koja zajednički ispunjava funkciju cjelokupnog računalnog sustava. Osnovna načela koja uvodi arhitektura zasnovana na uslugama je izlaganje programskih funkcionalnosti korištenjem otvorenih i standardima usvojenih tehnologija u obliku programskih usluga, mogućnost dinamičkog pronalaska, odabira i uključivanja programskih usluga u poslovni proces za vrijeme rada sustava, komunikacija programskih usluga zasnovana na razmjeni poruka te mogućnost povezivanja elementarnih usluga u složene usluge koje je moguće koristiti za daljnje povezivanje ili krajnje korištenje.

2.1.1 Programske usluge

Osnovni gradivni elementi programskog sustava zasnovanog na uslugama su programske usluge. Programske usluge su samostalne programske jedinice koje pružaju određenu vrstu dobro definirane programske funkcionalnosti. Funkcionalnosti koje programske usluge obavljaju i pružaju korisnicima protežu se od pružanja uslužnih

informacija, kao što su prognoza vremena ili dnevne vijesti, do obavljanja složenih procesa, kao što su upravljanje kadrovskom politikom poslovne organizacije, elektroničko poslovanje, udaljena znanstvena mjerjenja, matematički proračuni i slično. Programska usluga u potpunosti je određena pristupnom točkom računalne mreže na kojoj je dostupna, programskom funkcionalnoću koju pruža te pristupnim sučeljem putem kojeg se ostvaruje pristup njezinim funkcionalnostima. Usluge su fizički smještene kod pružatelja usluga koji osiguravaju njihovu dostupnost, korištenje i održavanje, a po potrebi i naplaćuju njihovo korištenje.



Slika 2.1 Struktura programske usluge

Pristup programskim uslugama ostvaruje se korištenjem otvorenih i standardima prihvaćenih programskih tehnologija. Prihvatanje standardnog skupa programskih tehnologija osnovni je preduvjet za omogućavanje povezivosti programskih usluga na globalnoj razini i prevladavanje tehnoloških i organizacijskih različitosti koje vladaju u različitim radnim okruženjima. Programska usluga se, stoga, može definirati kao bilo koji oblik programske funkcionalnosti izložen na korištenje putem otvorenih i standardom dogovorenih pristupnih sučelja. Izlaganje programskih funkcionalnosti putem otvorenih i standardima utvrđenih sučelja poduprto je razdvajanjem programskog ostvarenje jezgre poslovne logike i sučelja koje osigurava pristup do te logike. Slikom 2.1 prikazana je struktura programske usluge podijeljena na jezgrentu poslovnu logiku i omotač sa standardnim i otvorenim sučeljem. Jezgrena programska logika koja ostvaruje funkcionalnosti programske usluge može biti zasnovana na proizvoljnoj programskoj arhitekturi, može biti smještena na jednom mrežnom čvoru ili raspodijeljena na više njih,

može biti ostvarena primjenom proizvoljnih programskih jezika i izvoditi se u okolini s proizvoljnim operacijskim sustavom. U općem slučaju, različiti pružatelji usluga svoje programske funkcionalnosti ostvaruju koristeći raznorodne i međusobno neuskladive programske tehnologije. Sve dok se pristup do tih funkcionalnosti ostvaruje putem sučelja koja koriste otvorene i standardne komunikacijske protokole i tehnologije za objedinjavanje programske potpore (engl. *software integration*), programske usluge je moguće koristiti i povezivati na globalnoj razini.

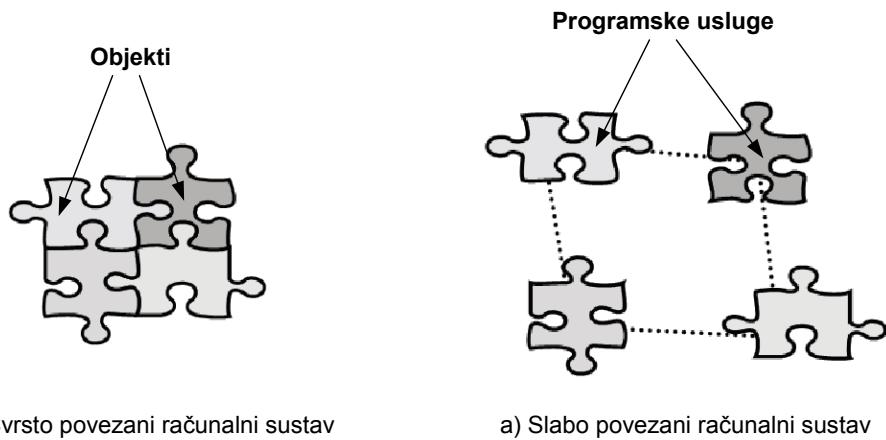
Razdvajanjem jezgre poslovne logike i pristupnih sučelja programskih usluga, olakšana je pretvorba postojećih naslijedenih programskih sustava (engl. *legacy systems*) koji su izgrađeni primjenom nestandardnih tehnologija svojstvenih pojedinoj proizvođaču programske potpore, u otvorene i standardizirane programske usluge. Naslijedeni sustavi ostvaruju jezgru poslovne logike programske usluge, dok se njihova dostupnost na globalnoj razini postiže izgradnjom odgovarajućih pristupnih sučelja zasnovanih na standardiziranim programskim tehnologijama. Zbog svoje raspodijeljenosti, neovisnosti o tehnologijama programskog ostvarenja te mogućnosti komunikacije i povezivanja s drugim uslugama dostupnih u globalnoj mreži Internet, programske usluge pretvaraju naslijedene i novoizgrađene programske sustave u globalnu programsku infrastrukturu za razvoj raspodijeljenih računalnih sustava.

2.1.2 Slabo povezivanje

Raspodijeljeni računalni sustavi izgrađeni na načelima arhitekture zasnovane na uslugama su slabo povezani računalni sustavi. Slaba povezanost računalnog sustava podrazumijeva postojanje stvarnih zavisnosti (engl. *real dependencies*) među sastavnim dijelovima sustava, dok su umjetne zavisnosti (engl. *artificial dependencies*) svedene na najmanju moguću mjeru. Stvarne zavisnosti među dijelovima sustava određene su njihovim funkcijskim zavisnostima. Umjetne zavisnosti određene su ograničenjima koja definiraju način na koji se ostvaruje suradnja sastavnih dijelova zbog njihovih stvarnih zavisnosti. Primjerice, stvarna zavisnost između korisnika usluge vijesti i novinske agencije izražena je u obliku sadržaja vijesti. Umjetna zavisnost između tih dvaju elemenata je način na koji novinska agencija korisniku omogućuje dostupnost sadržaja vijesti. Stvarne zavisnosti uvijek postoje i nije ih moguće ukloniti iz sustava. Umjetne zavisnosti također nije moguće u potpunosti ukloniti, ali njihovo postojanje mora biti svedeno na najmanju moguću mjeru. Primjerice, ako korisnik, kako bi došao do sadržaja vijesti, mora kupiti primjerak tiskanog izdanja novina, onda u sustavu postoji visok stupanj umjetne zavisnosti. Međutim, ako je

sadržaj vijesti korisniku ponuđen u tiskanom izdanju, na *web* stranici agencije te televizijskim i radio prijenosom, umjetne su zavisnosti znatno umanjene.

Razlika između čvrsto i slabo povezanih računalnih sustava istaknuta je slikom 2.2. Za sustave izgrađene primjenom objektno orientirane programske paradigmje svojstvena je čvrsta povezanost sastavnih dijelova, odnosno objekata. Veze među objektima ostvaruju se pozivanjem metoda nad objektima. Iako se promjene u unutrašnjoj gradi i načinu programskog ostvarenja pozvanog objekta neće odraziti na pozivajući objekt sve dok se ne mijenja sučelje pozvanog objekta, programsko ostvarenje pozivajućeg objekta ovisno je o izvedbi tog sučelja. Svi objekti koji su međusobno povezani moraju unaprijed znati za postojanje ostalih objekata te poznavati način i uvjete korištenja njihovih sučelja. Dinamičko povezivanje nepoznatih objekata za vrijeme rada sustava nije moguće. Čvrsto povezani sustav čini zbijenu programsku cjelinu u kojoj su svi sastavni dijelovi i način njihova međudjelovanja unaprijed poznati.



Slika 2.2 Načini povezivanja sastavnih dijelova računalnog sustava

Uklanjanjem umjetnih ovisnosti među programskim uslugama kao sastavnim dijelovima sustava zasnovanog na uslugama omogućen je dinamički odabir i uključivanje usluga u tijek izvođenja cjelokupnog programskog sustava. Odabir usluga čijim se povezivanjem postiže određena programska funkcionalnost moguće je određivati na zahtjev, za vrijeme rada sustava. Dinamičko povezivanje programskih usluga nije moguće ako usluge trebaju unaprijed poznavati način pristupa do programskih usluga s kojima se trebaju povezati. Povezivanje programskih usluga se, stoga, ne obavlja na osnovi čvrstog povezivanja pozivajuće usluge na programsko sučelje pozvane usluge, već se ostvaruje na osnovi opisa usluge (engl. *service description*). Svaka programska usluga opremljena je dokumentom kojim se opisuje način pristupanja do njezinih funkcionalnosti. Dokument s

opisom usluge sadrži naziv usluge, opis funkcionalnosti koje je sposobna obaviti, adresu pristupne točke u računalnoj mreži na kojoj je dostupna, podaci koje je usluzi potrebno proslijediti pri pozivu te podaci koje usluga vraća kao rezultat izvođenja. Dokument s opisom usluge objavljuje pozvana, a koristi pozivajuća programska usluga.

2.1.3 Komunikacija zasnovana na razmjeni poruka

Programske usluge međusobno komuniciraju razmjenom poruka. Poruke su strukture podataka kojima se parametri poziva usluge prenose od pozivajuće prema pozvanoj usluzi, a rezultati izvođenja pozvane usluge vraćaju pozivajućoj usluzi. U cilju omogućavanja razmjene podataka i suradnje među programskim uslugama izgrađenih primjenom raznorodnih programskih tehnologija, oblik zapisa poruka i njihova razmjena odvija se primjenom otvorenih i standardnih komunikacijskih protokola.

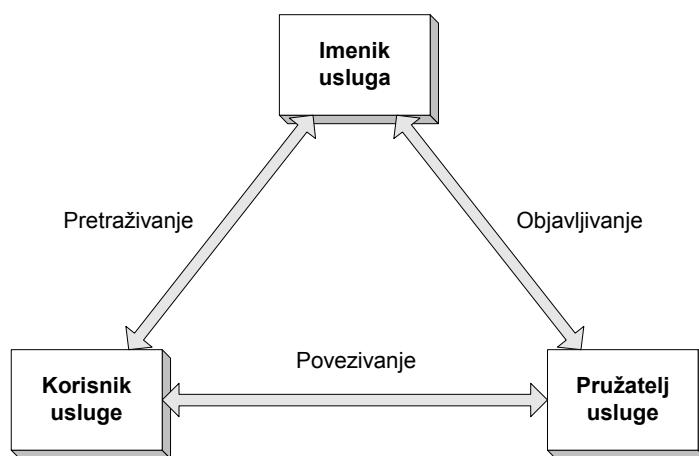
Komunikacija razmjenom poruka predstavlja bitan čimbenik pri izgradnji slabo povezanih računalnih sustava. Dio podataka sadržanih u dokumentu s opisom usluge sadrži pravila za oblikovanje ulaznih i izlaznih poruka programske usluge. Pozivajuća programska usluga koristi ta pravila kako bi mogla oblikovati poruku u obliku koji zahtijeva pozvana usluga. Mogućnost oblikovanja poruka na osnovi dokumenata s opisom usluga omogućuje izgradnju programskih usluga u potpunosti neovisnih o sučeljima ostalih usluga s kojima trebaju ostvarivati komunikaciju i suradnju. Tijekom izgradnje programske usluge, pažnja je usmjerena na ostvarivanje programske logike poslovnog procesa koji usluga obavlja te na ostvarivanje funkcijskih zavisnosti s drugim uslugama. Pristupanje sučeljima udaljenih usluga moguće je automatizirati i provoditi na osnovi trenutno važećih dokumenata s opisom programskih usluga.

2.1.4 Osnovne komponente arhitekture zasnovane na uslugama

Arhitektura zasnovana na uslugama uvodi i definira odnose između triju glavnih komponenata računalnog sustava zasnovanog na uslugama. Arhitekturu zasnovanu na uslugama čine pružatelj usluge (engl. *service provider*), korisnik usluge (engl. *service requestor*) i imenik usluga (engl. *service discovery registry*). Međudjelovanjem ovih triju komponenata omogućena je izgradnja raspodijeljenih, slabo povezanih računalnih sustava zasnovanih na uslugama. Odnosi između triju osnovnih komponenata arhitekture zasnovane na uslugama prikazani su slikom 2.3.

Međudjelovanje pružatelja, korisnika i imenika programskih usluga postiže se uporabom postupaka objavljivanja (engl. *publish*), pretraživanja (engl. *find*) i povezivanja (engl. *bind*). Pružatelj usluge udomljuje određene programske funkcionalnosti uobličene u

programske usluge. Programske funkcionalnosti u obliku programskih usluga dostupne su korisnicima putem javno izloženih pristupnih sučelja, ostvarenih korištenjem otvorenih i standardom usvojenih tehnologija. Za svaku programsku uslugu koju nudi, pružatelj usluge sastavlja dokument s opisom usluge (engl. *service description*). Dokument s opisom usluge pružatelj usluge objavljuje u imeniku usluga, čime on postaje dostupan korisnicima za javnu uporabu. Pretraživanjem imenika usluga, korisnik usluge pronalazi odgovarajuću uslugu i pribavlja njezin dokument s opisom usluge. Analizom dokumenta s opisom usluge, korisnik doznaće adresu pružatelja usluge, povezuje se s njim i nastavlja koristiti uslugu prema uputama sadržanima u tom dokumentu.

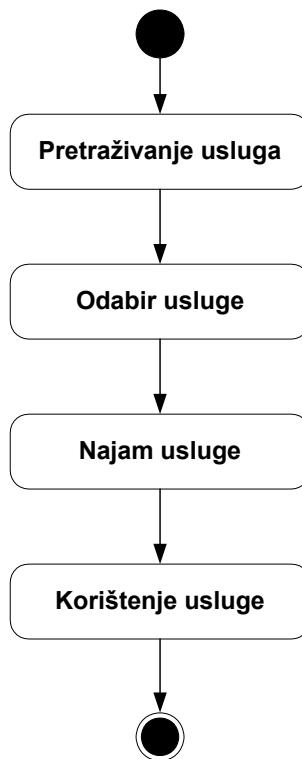


Slika 2.3 Osnovni dijelovi arhitekture zasnovane na uslugama i način njihova povezivanja

Pristup razvoju programskih sustava primjenom načela arhitekture zasnovane na uslugama razlikuje se od tradicionalnih pristupa koji koriste objektno orientiranu ili komponentnu paradigmu oblikovanja programske potpore. Tradicionalni pristup oblikovanju sastoji se od oblikovanja modela izvođenja poslovnog procesa, rastavljanja logike poslovnog procesa u odgovarajući broj objekata ili programskih komponenti, programskog ostvarenja pojedinih objekata ili komponenata u nekom od prikladnih programskih jezika te njihovog povezivanja u čvrstu cjelinu. Pri oblikovanju programskih sustava primjenom načela arhitekture zasnovane na uslugama, postupci pisanja programske logike za ostvarenje funkcionalnosti pojedinih objekata ili komponenata i njihovo povezivanje zamjenjuju se postupcima pretraživanja, odabira, najma i korištenja programskih usluga dostupnih na tržištu usluga.

Slikom 2.4 prikazan je postupak izgradnje programskih sustava zasnovanih na uslugama. Pretraživanjem tržišta usluga pronalaze se usluge koje obavljaju traženu funkcionalnost. Budući da je na tržištu usluga u općem slučaju ponuđen veći broj usluga koje

obavljaju istu ili slične funkcionalnosti, odluka o odabiru donosi se na osnovi njihovih nefunkcijskih svojstava. Nefunkcijska svojstva usluga uključuju kakvoću ponuđene usluge koja je mjerljiva kroz učinkovitost obavljanja posla, sigurnost korisnika pri korištenju usluge, cijenu korištenja usluge, ugled pružatelja usluge među korisnicima i slična svojstva bitna za provedbu poslovnog procesa. Nakon odluke o odabiru najpovoljnije usluge, korisnik unajmljuje uslugu od pružatelja usluge te je koristi u svrhu provođenja vlastitog poslovnog procesa.



Slika 2.4 Pristup izgradnji programskih sustava primjenom načela arhitekture zasnovane na uslugama

2.1.5 Povezivanje usluga

Ključna uloga u izgradnji računalnih sustava zasnovanih na uslugama pripada postupku povezivanja programskih usluga (engl. *service composition*) [13]. Povezivanje usluga omogućuje stvaranje novih vrijednosti na osnovi postojećih dobara. Povezivanjem usluga, iskorištavaju se funkcionalnosti koje pružaju postojeće usluge te se one okupljaju i povezuju u složene usluge. Različitim načinima povezivanja postojećih usluga omogućava se stvaranje novih usluga bez potrebe za njihovim razvijanjem. Novonastale složene usluge moguće je koristiti u daljim postupcima povezivanja ili ih je moguće ponuditi krajnjim korisnicima. Pružatelji složenih programskih usluga koji korisnicima nude usluge ostvarene postupkom povezivanja usluga ponuđenih od strane drugih pružatelja nazivaju se

sastavljačima usluga (engl. *service aggregators*). Sastavljači usluga postaju uobičajeni pružatelji usluga objavljivanjem dokumenata s opisom složenih programskih usluga u imeniku usluga.

Osim pružanja složene programske usluge, zadatak sastavljača usluga je briga o osiguravanju kakvoće i sigurnosti složene usluge i njezinih korisnika. Kakvoća složene programske usluge ovisi o kakvoći pojedinačnih usluga od kojih je sastavljena. Zadatak sastavljača usluga je redovito praćenje nefunkcijskih svojstava pojedinačnih usluga od kojih je sastavljena složena usluga, pronalaženje zamjenskih usluga iste ili sličnih funkcionalnosti te uključivanje onih usluga u sastav složene usluge kojima se postiže prihvatljiva razina ukupne kakvoće. Mjerila kakvoće ponuđene usluge uglavnom obuhvaćaju cijenu korištenja, učinkovitost, brzinu te stupanj dostupnosti, pouzdanosti i vjerodostojnosti pružene usluge.

Pitanja sigurnosti složenih usluga i njihovih korisnika uglavnom se odnose na prava korištenja pojedinačnih usluga od strane pojedinih korisnika složene usluge te na osiguravanje privatnosti korisnika i pojedinačnih usluga koje su u sastavu složene usluge. Korisnička prava kod pružatelja složenih usluga i pružatelja pojedinačnih usluga u sastavu složene usluge mogu se bitno razlikovati. Stoga, pružatelj složene usluge mora zabraniti korištenje složene usluge onim korisnicima kojima je zabranjeno korištenje neke od pojedinačnih usluga. Druga je mogućnost upravljanja pravima pristupa sklapanje dogovora s pružateljima pojedinačnih usluga, kojim se svim korisnicima koji imaju pravo korištenja složene usluge osigurava pristup do njezinih pojedinačnih sastavnica. Drugi bitan čimbenik sigurnosti složenih usluga i njihovih korisnika je privatnost. Korištenje složene programske usluge ne bi trebalo odavati informacije o pojedinačnim uslugama od kojih se ona sastoji, kao što ni pojedinačne programske usluge ne bi trebale dolaziti u posjed podataka o korisnicima složene usluge. Korisnici složene programske usluge koriste složenu uslugu u svoje ime i svoju osobnu korist, a pružatelj složene usluge koristi pojedinačne usluge u svoje ime, ali u korist svojih korisnika.

2.1.6 Tehnologije i standardi za izgradnju sustava zasnovanih na uslugama

Izgradnja sustava zasnovanih na uslugama poduprta je skupom otvorenih i standardima usvojenih tehnologija kojima se omogućava objavljivanje, pretraživanje, objedinjavanje i povezivanje programskih usluga. Prevladavajući skup standardnih tehnologija i protokola okupljen je u radni okvir pod nazivom *Web Services* [14, 15]. *Web Services* radni okvir predstavlja infrastrukturu koja omogućuje međudjelovanje, povezivanje i objedinjavanje programskih sustava različitih proizvođača zasnovanih na raznorodnim

programskim tehnologijama. Razvoj sustava zasnovanih na uslugama uz primjenu *Web Services* radnog okvira zasniva se na korištenju *XML*, *SOAP*, *WSDL* i *UDDI* standarda.

XML (engl. *Extensible Markup Language*) [18, 19] je otvoren i standardom usvojen jezik za razmjenu podataka u otvorenim mrežnim okruženjima. Koristi se za predstavljanje strukturiranih vrsta podataka. Opis podatkovnih struktura ostvaren elementima *XML* jezika u potpunosti se zasniva na tekstualnom zapisu, što ga čini neovisnom tehnologijom za prikaz podataka u različitim programskim okolinama. *XML* jezikom opisuju se strukture podataka, ali se ne definira njihovo značenje. Tumačenje značenja podatkovnih struktura predstavljenih *XML* jezikom ostavljeno je pojedinim područjima primjene.

Jedno od područja primjene *XML* jezika je *SOAP* protokol [20]. *SOAP* protokol je komunikacijski protokol logičke razine koji se koristi za razmjenu podataka među programskim uslugama. *SOAP* protokolom definirana su pravila oblikovanja poruka kojima se između pozivajuće i pozvane programske usluge prenose parametri poziva i rezultati izvođenja. Za prijenos *SOAP* poruka računalnom mrežom moguće je koristiti proizvoljni fizički prijenosni protokol. Najčešće korišten prijenosni protokol je *HTTP* (engl. *HyperText Transfer Protocol*) protokol zbog njegovih dobrih svojstava u pogledu otvorenosti, prihvaćenosti i propusnosti od strane sigurnosnih sustava u većini poslovnih organizacija.

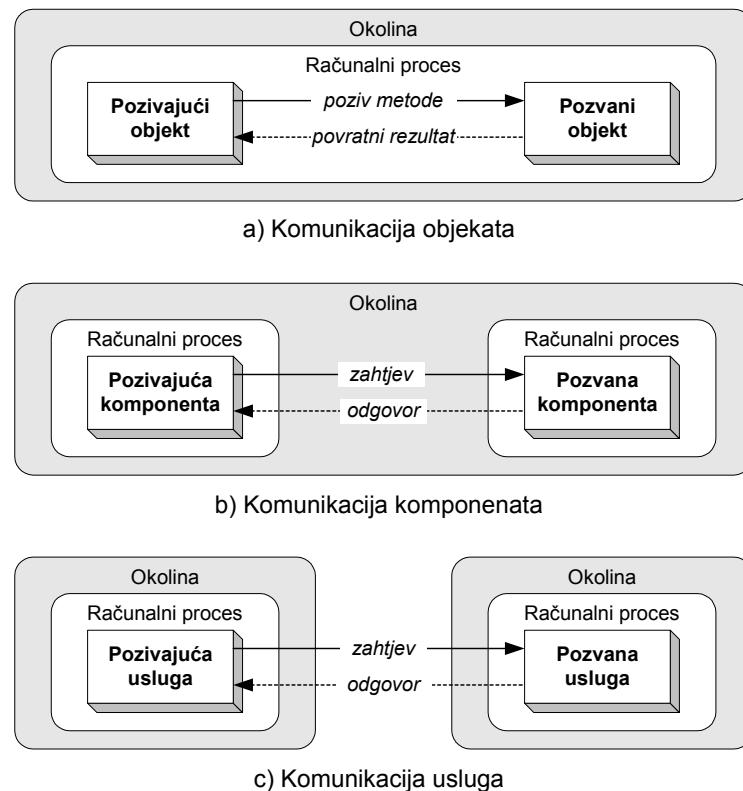
WSDL (engl. *Web Service Description Language*) [21] je standardom propisani jezik za opisivanje sučelja programskih usluga. Zasnovan je na *XML* jeziku, a koristi se za objavljivanje podataka o pristupnim točkama programskih usluga i načinu korištenja njihovih pristupnih sučelja. *WSDL* dokumenti objavljaju se u imeniku usluga. Na osnovi podataka sadržanih u *WSDL* dokumentu, pozivajuće programske usluge su u stanju oblikovati *SOAP* poruku u skladu s pravilima nametnutim od strane pozvane usluge. Uporaba *WSDL* dokumenata pri oblikovanju *SOAP* poruka omogućuje automatizirano i dinamičko povezivanje usluga, što predstavlja osnovicu za izgradnju slabo povezanih sustava.

Objavljivanje *WSDL* dokumenata u imeniku usluga, pretraživanje imenika usluga te dohvaćanje *WSDL* dokumenata iz imenika usluga obavlja se primjenom *UDDI* (engl. *Universal Description, Discovery, and Integration*) protokola [22]. Kao i sve dosadašnje tehnologije okupljene u *Web Services* radni okvir, i *UDDI* standard za opis svojih podatkovnih struktura koristi *XML* jezik, dodajući elementima *XML* jezika odgovarajuća značenja. Uloga *UDDI* standarda i protokola slična je ulozi žutih, zelenih i bijelih stranica u funkciji poslovnog oglašavanja. *UDDI* standardom definirana su pravila izgradnje imenika

usluga u obliku programske usluge te *UDDI* protokol kojim se propisuje način međudjelovanja korisnika i pružatelja usluga s imenikom usluga.

2.2 Usporedba oblikovanja zasnovanog na uslugama s oblikovanjem zasnovanim na objektima i komponentama

Uvođenjem paradigme oblikovanja i razvoja raspodijeljenih računalnih sustava zasnovanih na uslugama, objektno orijentirano oblikovanje i oblikovanje zasnovano na komponentama nisu potisnuti. Svaka od triju programskih paradigmi razvoja računalnih sustava ima svoje vlastito područje primjene. Odabir programske paradigme ovisi o svojstvima programskog sustava koji je potrebno izgraditi te uvjetima koji vladaju u okolini u kojoj se izgrađeni sustav izvodi.



Slika 2.5 Usporedba međudjelovanja objekata, komponenti i usluga

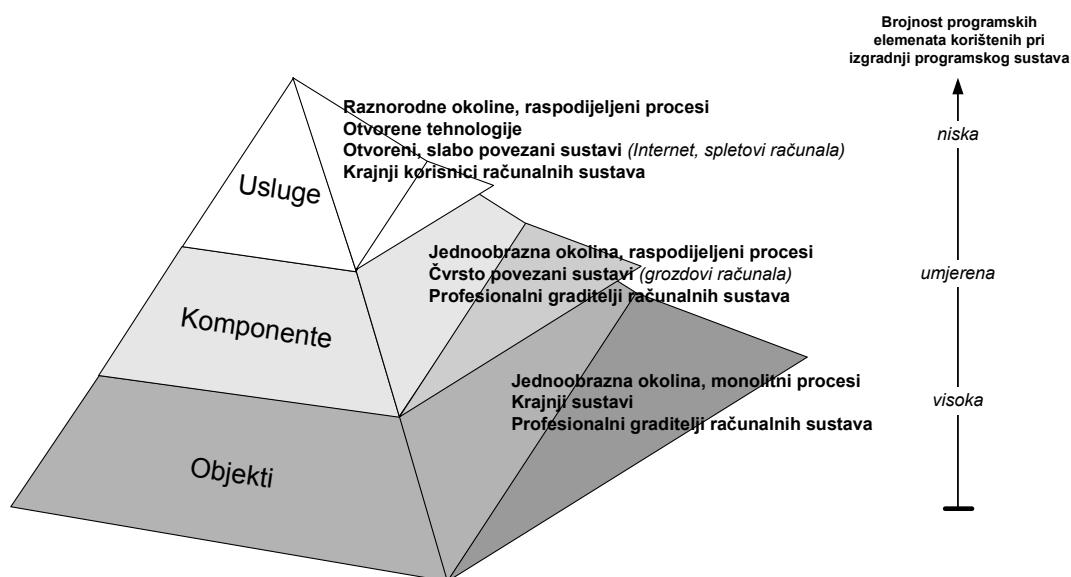
Osnovna razlika za vrijeme izvođenja programskih sustava izgrađenih primjenom triju programskih paradigmi proizlazi iz položaja pozivajućeg programskog elementa u odnosu na pozvani element te računalne okoline u kojoj se pozivajući i pozvani elementi nalaze i izvode [23]. U računalnim sustavima koji su izgrađeni primjenom objektno orijentirane programske paradigme, funkcionalnosti pozivajućeg i pozvanog objekta izvode se unutar

istog računalnog procesa. Budući da se jedan računalni proces ne može istovremeno izvoditi u više računalnih okolina, oba su objekta ujedno i dijelovi iste okoline. Ako su pozivajući i pozvani programski elementi dijelovi različitih računalnih procesa, odabir programske paradigme ovisi o okolinama u kojima se ti procesi izvode. Ako se oba procesa izvode u istoj računalnoj okolini, moguće je upotrijebiti paradigmu oblikovanja zasnovanu na komponentama. Povezivanje raspodijeljenih programskih komponenata je moguće provesti sve dok su one izgrađene istom programskom tehnologijom i izvode se u istim računalnim okolinama. Ako se dva procesa izvode u raznorodnim računalnim okolinama, povezivanje programskih elemenata moguće je provesti jedino uporabom programske paradigme zasnovane na uslugama. Za objedinjavanje raznorodnih programskih okolina potrebno je koristiti otvorene i standardima usvojene tehnologije, što predstavlja okosnicu arhitekture zasnovane na uslugama. Usporedni prikaz mogućnosti povezivanja programskih funkcionalnosti u odnosu na njihov položaj u računalnim procesima i računalnu okolinu u kojoj se ti procesi izvode prikazan je slike 2.5.

Na osnovi razmatranja mogućnosti međudjelovanja programskih elemenata prikazanog slike 2.5 dolazi se do zaključka da je oblikovanje zasnovano na uslugama najopćenitiji način objedinjavanja programskih funkcionalnosti. Oblikovanje zasnovano na uslugama moglo bi zamijeniti preostale dvije programske paradigme oblikovanja računalnih sustava. Odabir programske paradigme, međutim, ima značajne posljedice na krajnja svojstva učinkovitosti izgrađenog sustava glede brzine rada te opterećenja računalnih i komunikacijskih sredstava. Računalni sustav pokazuje najveći stupanj učinkovitosti ako je izgrađen primjenom objektno orientirane programske paradigme. Izravnom komunikacijom objekata postiže se najveće brzine odziva sustava i najmanja opterećenja računalnih sredstava jer se pozivanje metoda nad objektima odvija unutar istog računalnog procesa. Prilikom pozivanja programske komponente poruke s pozivima moraju izlaziti izvan granica procesa. Umjesto izravnog poziva metoda, komunikacija između komponenata odvija se odgovarajućim komunikacijskim protokolima koji na osnovi parametara poziva oblikuju poruke te ih prenose između dvaju procesa primjenom nekog od modela međuprocesne komunikacije. Međuprocesna komunikacija u sustav unosi znatna kašnjenja i dodatna opterećenja računalnih, a u slučaju raspodijeljenih procesa, i mrežnih sredstava. Komunikacija među procesima koji se izvode unutar iste okoline je, međutim, još uvek znatno brža od one koja se odvija između procesa koji se izvode u različitim okolinama. Procesi unutar iste okoline uglavnom koriste komunikacijske protokole koji su optimirani i prilagodeni upravo danoj okolini. Procesi u različitim okolinama za međusobnu komunikaciju moraju koristiti komunikacijske protokole koji su rezultat dogovora obiju

strana ili su usvojeni općim standardima. Komunikacija programskih usluga je, stoga, najopćenitiji, ali i najneučinkovitiji način objedinjavanja programskih funkcionalnosti. Otvorenost i mogućnost globalne povezivosti sustava zasnovanih na uslugama plaćeni su smanjivanjem njihove učinkovitosti.

Razlike u mogućnostima primjene pojedinih programskih paradigma koje proizlaze iz položaja računalnih procesa i vrste računalnih okolina određuju i područje primjene računalnih sustava izgrađenih primjenom pojedine programske paradigmme. Osim po svojstvima učinkovitosti i područjima primjene, tri paradigmme razvoja računalnih sustava razlikuju se i po prikladnosti za korištenje od strane pojedinih graditelja programskih sustava. Slikom 2.6 prikazana je piramida triju paradigm oblikovanja računalnih sustava s istaknutim područjima primjene i očekivanim sposobnostima korisnika svake od njih.



Slika 2.6 Piramida programskih paradigm za izgradnju računalnih sustava

Osnovica piramide pripada objektno orijentiranoj paradigmri razvoja programskih sustava. Objektno orijentirana paradigmra oblikovanja primjerena je razvoju programske potpore za krajnje sustave (engl. *end systems*). Krajnji sustavi ostvaruju nedjeljive programske funkcionalnosti koje nemaju svojstvo raspodijeljenosti. Čitav krajnji sustav sastoji se od jednog računalnog procesa koji se izvodi na jednom mjestu u mreži. Budući da je objektno orijentirana paradigmra oblikovanja uglavnom usmjerena na programsko ostvarenje funkcionalnosti pojedinih razreda, njena primjena iziskuje duboko razumijevanje načela rada računalnih sustava i objektno orijentiranih programskih jezika. Ova je

programska paradigma namijenjena uporabi od strane profesionalnih graditelja računalnih sustava.

Središnji dio tehnologijske piramide zauzima oblikovanje programske potpore zasnovano na komponentama. Ova je programska paradigma pogodna za izgradnju raspodijeljenih sustava sve dok su oni ostvareni primjenom istih programskih tehnologija i dok se izvode u istim radnim okolinama. Računalni sustavi takvih svojstava su čvrsto povezani raspodijeljeni sustavi u obliku grozdova računala (engl. *clusters*) [24]. Grozdovi računala su raspodijeljeni računalni sustavi koji se koriste za izvođenje primjenskih programa visoke učinkovitosti za koje je potrebna iznimno velika računalna snaga. Komunikacija i međudjelovanje komponenata računalnog grozda moraju biti optimirani u što većoj mjeri, što nužno dovodi do njihova čvrstog povezivanja. Uporaba oblikovanja zasnovanog na komponentama također zahtjeva dobro poznавanje računalnih tehnologija, komunikacijskih protokola i programskih jezika te je ova programska paradigma također namijenjena uporabi od strane profesionalnih graditelja računalnih sustava.

Vrh piramide programskih paradigm zauzima oblikovanje programske potpore zasnovano na uslugama. Otvorene i standardima usvojene programske tehnologije za objedinjavanje programskih usluga u poslovne procese omogućuju izgradnju računalnih sustava velikih razmjera (engl. *large scale*), čiji su sastavni dijelovi zemljopisno rasprostranjeni, a izvode se u raznorodnim računalnim okolinama. Računalni sustavi takvih svojstava nazivaju se spletovima računala (engl. *Grid*) [25]. Glavnina napora graditelja programske potpore primjenom smjernica arhitekture zasnovane na uslugama usmjeren je na odabir programskih usluga ponuđenih na tržištu i način njihova povezivanja i usklajivanja njihova rada. Budući da su programske tehnologije koje podupiru razvoj sustava zasnovanih na uslugama otvorene i standardima usvojene, postupke pretraživanja, pronalaska, povezivanja i objedinjavanja programskih usluga moguće je automatizirati izgradnjom prikladnih i za uporabu jednostavnih programskih alata. Zbog tih svojstava, oblikovanje zasnovano na uslugama omogućava brži i jednostavniji način izgradnje složenih programskih sustava, čiji graditelji ne moraju nužno biti školovani stručnjaci, već mogu biti i krajnji korisnici. Računarstvo zasnovano na uslugama izlazi iz okvira industrije profesionalne programske potpore, omogućavajući krajnjim korisnicima razvoj vlastitih programskih sustava koji ispunjavaju njihove trenutne zahtjeve.

Slikom 2.6 istaknuta je i brojnost upotrijebljenih programskih elemenata pri izgradnji programskih sustava primjenom različitih paradigm oblikovanja. Kretajući se od dna prema vrhu piramide programskih paradigm, brojnost programskih elemenata kreće se od visoke,

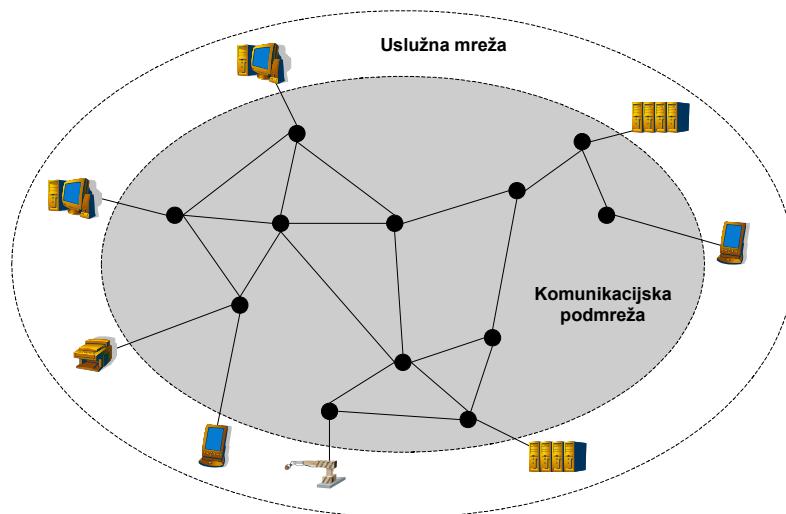
preko umjerene, do vrlo niske. Programskoj paradigmi zasnovanoj na objektno-orientiranom oblikovanju svojstvena je visoka brojnost upotrijebljenih programskih elemenata, odnosno objekata. Budući da je programskim sustavima koji su izgrađeni primjenom objektno orijentirane programske paradigme svojstvena sitna zrnatost (engl. *fine grained systems*), za njihovo je programsko ostvarenje potreban veliki broj objekata. Izrazito visoka brojnost objekata posljedica je i primjene objektno orijentirane paradigme oblikovanja za ostvarenje unutarnje programske logike komponenata i usluga. Programske komponente ostvarene su međudjelovanjem odgovarajućeg broja objekata te je njihov broj znatno manji od broja korištenih objekata. Programske komponente povezane u programski sustav koji pruža zaokruženu programsku funkcionalnost koriste se za izgradnju programskih usluga. Sustavima zasnovanim na uslugama svojstvena je krupna zrnatost (engl. *coarse grained systems*), što znači da je brojnost programskih usluga izrazito niska u odnosu na brojnost objekata i programskih komponenata.

Izgradnja raspodijeljenih sustava primjenom smjernica računarstva zasnovanog na uslugama prepoznata je od strane industrije za razvoj informacijskih tehnologija i sustava (engl. *information technology industry, IT industry*) kao paradigma koja omogućuje objedinjavanje (engl. *integration*) raznorodnih programskih sustava, kako novoizgrađenih, tako i onih naslijedenih (engl. *legacy systems*). Mogućnost objedinjavanja raznorodnih programskih sustava osnovni je preduvjet za njihovo povezivanje na globalnoj razini putem mreže Internet. Mogućnost globalnog povezivanja i objedinjavanja raznorodnih računalnih sustava osnovni su razlozi odabira oblikovanja zasnovanog na uslugama kao paradigme za razvoj suvremenih raspodijeljenih programskih sustava.

3 Prividne logičke mreže

Infrastruktura globalne mreže Internet poduprta *IP* (engl. *Internet Protocol*) skupinom komunikacijskih protokola osigurava globalnu povezivost računalnih sustava raspodijeljenih diljem svijeta. Mreža Internet doživjela je iznimno visok stupanj prihvaćenosti od strane graditelja mrežne programske potpore i korisnika mrežnih usluga zbog dobro definiranog i standardom prihvaćenog osnovnog skupa komunikacijskih protokola okupljenih u *IP* skupinu protokola. Programske usluge izgrađene u mreži Internet ostvaruju programsku logiku svojstvenu pojedinom području primjene, dok istovremeno iskorištavaju zajedničku mrežnu infrastrukturu za ostvarivanje međusobne komunikacije.

Infrastruktura mreže Internet podijeljena je na uslužnu mrežu i komunikacijsku podmrežu. Uslužna mreža (engl. *service network*) sastavljena je od poslužitelja mrežnih usluga i korisničkih pristupnih uređaja koji koriste usluge poslužitelja. Komunikacijska podmreža (engl. *subnet*) upravlja oblikom i ustrojem mrežnih komunikacijskih putova, osiguravajući na taj način povezivost i komunikaciju čvorova uslužne mreže. Slikom 3.1 prikazana je podjela mreže Internet na uslužnu mrežu i komunikacijsku podmrežu. Uslužna mreža i komunikacijska podmreža sastoje se od čvorova i komunikacijskih veza među tim čvorovima. Čvorovi su mrežni elementi s mogućnošću obrade podataka, dok su komunikacijske veze elementi s mogućnošću prijenosa podataka. Čvorovi uslužne mreže obavljaju obradu podataka svojstvenu području primjene mrežne usluge koju pružaju, dok čvorovi komunikacijske podmreže provode postupak usmjeravanja podataka (engl. *routing*) između izvorišnih i odredišnih čvorova uslužne mreže.



Slika 3.1 Podjela mreže Internet na uslužnu mrežu i komunikacijsku podmrežu

Naslovljavanje čvorova uslužne mreže obavlja se dodjeljivanjem mrežnih adresa. Svakom čvoru uslužne mreže dodijeljena je jedinstvena mrežna adresa. Na osnovi mrežnih adresa, komunikacijska podmreža određuje položaj čvorova unutar uslužne mreže te usmjerava poruke od izvorišnih prema odredišnim čvorovima. Usmjeravanje poruka među čvorovima uslužne mreže obavljaju čvorovi komunikacijske podmreže. Prijenos podataka od izvorišnog do odredišnog čvora uslužne mreže u potpunosti je upravljan od strane komunikacijske podmreže te je neovisan o uslužnoj mreži, kao i vrsti i namjeni pojedine mrežne usluge. Organizacija ustroja uslužne mreže i postupaka usmjeravanja mrežnog prometa zasnovanog na mrežnim adresama čvorova uslužne mreže naziva se čvorovima potaknuto upravljanje oblikom i ustrojem mreže (engl. *node-centric network topology*) [26].

Upravljanje oblikom i ustrojem mreže Internet potaknuto mrežnim adresama čvorova uslužne mreže donedavno je zadovoljavalo većinu potreba mrežnih programskih sustava. Neovisnost komunikacijske podmreže o području primjene pojedinog mrežnog programskega sustava značajno je olakšavalo njihovu izgradnju jer je glavnina pažnje tijekom izgradnje mrežnih programskih sustava bila usmjerena na ostvarivanje poslovne logike svojstvene području primjene. S druge strane, upravljanje komunikacijskom podmrežom bilo je znatno pojednostavljeno jer je raznorodnost na razini uslužne mreže bila prikrivena malim skupom jezgrenih komunikacijskih protokola, algoritama i upravljačkih postupaka za prijenos podataka i upravljanje ustrojem mreže.

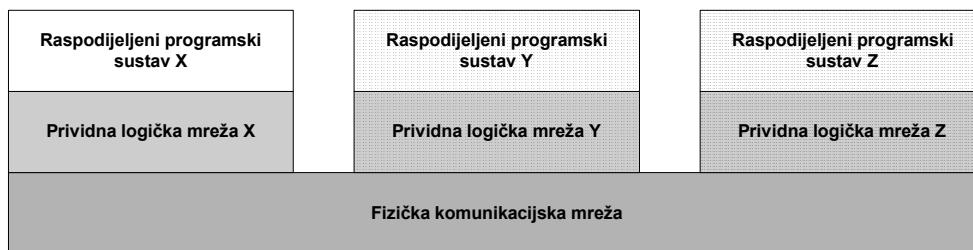
Razvojem raspodijeljenih programskih sustava s vremenski promjenjivim brojem čvorova, programskih sustava raspodijeljenih po pokretnim mrežnim čvorovima, bežičnih osjetilnih mreža (engl. *wireless sensor networks*) [27, 28] i računarstva zasnovanog na spletu računala (engl. *Grid computing*) [25, 29] pojavila se potreba za naprednjim načinima ostvarivanja mrežne komunikacije. Napredni mrežni programski sustavi zahtijevaju komunikacijsku infrastrukturu prilagođenu području primjene za koju su namijenjeni. Poslovna logika mrežnih programskih sustava i način upravljanja mrežnom komunikacijom približeni su izgradnjom prividnih logičkih mreža (engl. *virtual networks*). Prividne logičke mreže su programski sustavi za ostvarivanje mrežne komunikacije izgrađeni na postojećoj infrastrukturi mreže Internet te posebno prilagođeni primjeni od strane programskih sustava za koje su izgrađeni.

Raspodijeljeni programski sustavi izgrađeni nad prividnim logičkim mrežama ne koriste izravno komunikacijsku podmrežu mreže Internet, već za ostvarivanje komunikacije među svojim raspodijeljenim dijelovima iskorištavaju prividni komunikacijski prostor logičke mreže. Stvarna se komunikacija i dalje odvija posredstvom komunikacijske

podmreže. Prividne logičke mreže su zato po svojoj funkcionalnosti vrsta komunikacijskog programskog međusloja (engl. *communication middleware*) kojim se ostvaruje razdvajanje logičkog komunikacijskog prostora za izgradnju raspodijeljenih programskih sustava od stvarne komunikacijske mreže kojom se ostvaruje fizički prijenos podataka.

3.1 Komunikacijski podsustav prividne logičke mreže

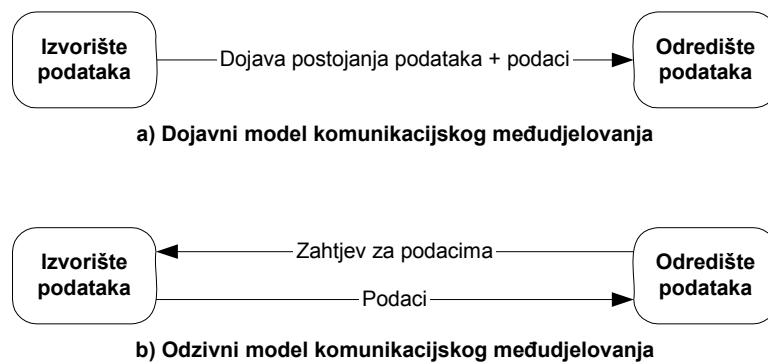
Raspodijeljeni programski sustav zasnovan na prividnoj logičkoj mreži sastoji se od tri osnovne funkcionalne cjeline [30], prikazane slikom 3.2. Fizička komunikacijska mreža ostvaruje funkcionalnosti stvarnog fizičkog prijenosa podataka. Prividna logička mreža prilagođava fizički komunikacijski prostor logičkom komunikacijskom prostoru svojstvenom području primjene raspodijeljenog programskog sustava. Konačno, raspodijeljeni programski sustavi izgrađuju se i raspodijeljuju u komunikacijskom prostoru prividne logičke mreže. Koristeći se zajedničkom fizičkom komunikacijskom infrastrukturom, različiti raspodijeljeni programski sustavi mogu oblikovati logički odvojene prividne mreže.



Slika 3.2 Osnovne funkcionalne cjeline raspodijeljenog programskog sustava zasnovanog na prividnoj logičkoj mreži

Osim prilagođavanja komunikacijskog prostora potrebama raspodijeljenih programskih sustava, prividne logičke mreže od raspodijeljenih programskih sustava skrivaju tehnologije korištene za fizički prijenos podataka. Fizička mreža ne mora nužno biti mreža Internet zasnovana na tehnologiji prijenosa podataka poduprtoj *IP* protokolom. Suvremeni raspodijeljeni programski sustavi vrlo često koriste spoj raznorodnih tehnologija fizičkog prijenosa podataka, primjerice spoj mreže Internet, različitih vrsta bežičnih mreža i telekomunikacijskih mreža. Izgradnjom prividne logičke mreže iznad raznorodnih fizičkih mreža te raspodjeljom funkcionalnosti programskih sustava unutar jedinstvenog logičkog komunikacijskog prostora, postupak izgradnje raspodijeljenih programskih sustava oslobođen je brige o tehnološkoj raznorodnosti prijenosa podataka na fizičkoj razini.

Modeli komunikacijskog međudjelovanja između komunicirajućih elemenata prividne mreže slični su postojećim modelima komunikacije među čvorovima fizičke mreže. Ovisno o načinu pribavljanja podataka, izvorišni i odredišni komunicirajući elementi mogu biti povezani dojavnim (engl. *proactive*) ili odzivnim (engl. *reactive*) modelom međudjelovanja [31]. Način rada dvaju osnovnih modela međudjelovanja te ključne razlike u njihovom ostvarenju prikazani su slikom 3.3.



Slika 3.3 Modeli komunikacijskog međudjelovanja elemenata prividne logičke mreže

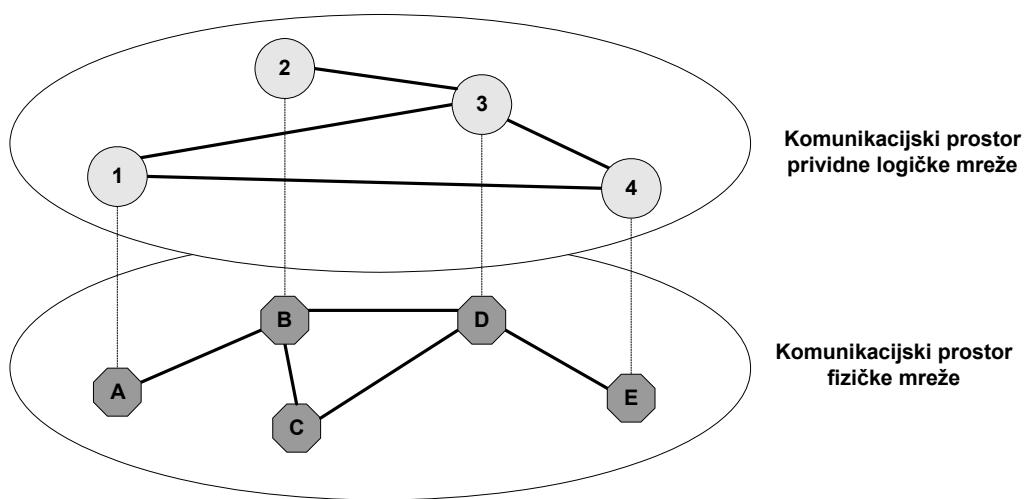
Dojavni model međudjelovanja koristi postupak razmjene podataka potaknut izvorištem podataka. U trenutku kada dođe u posjed podataka, izvorište podataka dojavljuje postojanje podataka svim odredištima za koja postoje pokazatelji zanimanja za tu vrstu podataka. Zanimanje za određenu vrstu podataka može izraziti odredište posebnim postupkom prijave pri izvorištu ili izvorište samostalno određuje skup odredišta kojima šalje dojavu o postojanju podataka. Za razliku od dojavnog modela, odzivni model međudjelovanja koristi postupak razmjene podataka potaknut odredištem podataka. Svaki put kada odredište podataka nastoji pribaviti podatke od izvorišta, šalje se poruka sa zahtjevom za podacima. Ako izvorište raspolaže traženim podacima, podaci se šalju odredištu u povratnoj poruci.

Oba modela međudjelovanja imaju određenih prednosti i nedostaka zbog kojih su primjenjivi pod određenim okolnostima. Primjena dojavnog modela pogodna je u slučajevima kada su trenuci raspoloživosti podataka na izvorištu nepoznati ili nepredvidljivi. Dojavni se model, stoga, primjenjuje za dojavljivanje postojanja novih sredstava ili članova unutar prividne mreže te za usmjeravanje sadržaja među članovima mreže. Prednost dojavnog modela je znatno manje opterećenje komunikacijskih veza jer je za razmjenu podataka potrebno razmijeniti samo jednu poruku između izvorišta i odredišta. Korištenje odzivnog modela s učestalim ispitivanjem raspoloživosti podataka u takvim bi okolnostima

bilo iznimno neučinkovito s gledišta opterećenja komunikacijskih veza, ali i izvorišta i odredišta podataka. Odzivni model primjenjuje se za pozivanje određene usluge izvorišta ili traženje podataka na zahtjev. S gledišta programskog ostvarenja, odzivni model je jednostavniji jer se razmjena podataka obavlja na zahtjev odredišta. U dojavnom modelu izvorište podataka mora koristiti dodatne postupke za određivanje skupa odredišta kojima je potrebno dojaviti postojanje raspoloživih podataka.

3.1.1 Upravljanje oblikom i ustrojem prividne mreže

Prividna mreža određena je rasporedom svojih komunicirajućih elemenata i načinom ostvarivanja komunikacije među tim elementima. Komunicirajući elementi prividne logičke mreže nazivaju se logičkim čvorovima. Logički čvorovi povezani su prividnim komunikacijskim vezama. Raspored logičkih čvorova prividne mreže po čvorovima fizičke mreže zajedno sa skupom prividnih komunikacijskih veza među logičkim čvorovima određuju oblik i ustroj prividne logičke mreže (engl. *virtual network topology*). Oblik i ustroj prividne logičke mreže prilagođen je potrebama raspodijeljenog programskog sustava za koji je prividna mreža izgrađena te je neovisan o obliku i ustroju fizičke mreže. Slikom 3.4 prikazan je raspored logičkih čvorova i logičkih komunikacijskih veza prividne mreže u odnosu na čvorove i komunikacijske veze fizičke mreže nad kojom je izgrađena prividna mreža.



Slika 3.4 Komunikacijski prostor prividne logičke mreže izgrađen u komunikacijskom prostoru fizičke mreže

Uspostavljanjem prividnih komunikacijskih veza među čvorovima prividne mreže oblikuje se korisnički definirani logički komunikacijski prostor koji najbolje odgovara potrebama raspodijeljenog programskog sustava. Rasprostiranje prividne mreže u

komunikacijskom prostoru fizičke mreže obuhvaća podskup čvorova i komunikacijskih veza fizičkog komunikacijskog prostora. Prividna komunikacijska veza kojom su izravno povezana dva logička čvora prividne mreže u općem se slučaju sastoji od većeg broja fizičkih veza među stvarnim fizičkim čvorovima. Putovanje poruke prividnom mrežom slijedeći jednu prividnu komunikacijsku vezu u fizičkoj se mreži odvija putovanjem poruke putem više fizičkih komunikacijskih veza. Primjerice, putovanje poruke u prividnom komunikacijskom prostoru od logičkog čvora 1 do logičkog čvora 4 slijedi izravnu prividnu komunikacijsku vezu između tih dvaju logičkih čvorova. U fizičkom komunikacijskom prostoru, međutim, ista poruka mora prolaziti putem tri ili četiri komunikacijske veze, ovisno o tome da li se usmjeravanje poruke od fizičkog čvora A do fizičkog čvora E obavlja komunikacijskim putem ABDE ili komunikacijskim putem ABCDE.

Upravljanje oblikom i ustrojem prividne mreže obuhvaća postupke i algoritme za raspoređivanje logičkih čvorova po fizičkim čvorovima, uspostavu logičkih komunikacijskih veza među logičkim čvorovima te usmjeravanje podataka od izvorišnog do odredišnog logičkog čvora. Budući da korisnik ili programski podsustav zadužen za upravljanje oblikom i ustrojem prividne logičke mreže nema u potpunosti dostupne podatke o obliku i ustroju fizičke mreže, moguća su znatna neslaganja između stvarnog i optimalnog oblika i ustroja prividne u odnosu na fizičku mrežu. Primjerice, u sustavu prikazanom slikom 3.4 ne postoji izravna prividna komunikacijska veza između logičkih čvorova 1 i 2. Logički čvor 1 šalje poruke logičkom čvoru 2 posredstvom logičkog čvora 3. U fizičkom komunikacijskom prostoru to uzrokuje put poruke od čvora A do čvora B, zatim od čvora B do čvora D te konačno od čvora D do čvora B. Iako je već nakon prolaska prvom komunikacijskom vezom poruka stigla na odredište, zbog neoptimanog ustrojstva prividnog komunikacijskog prostora poruka je putovala putem triju fizičkih komunikacijskih veza.

3.1.2 Naslovljavanje logičkih čvorova

Naslovljavanje logičkih čvorova predstavlja bitan čimbenik u oblikovanju prividne logičke mreže. Slično upravljanju oblikom i ustrojem prividne mreže, sustav naslovljavanja logičkih čvorova također je neovisan o sustavu imenovanja čvorova fizičke mreže. Način naslovljavanja čvorova prividne mreže prilagođen je potrebama raspodijeljenog programskog sustava. Svojstva koja sustav naslovljavanja logičkih čvorova mora zadovoljavati su jedinstvenost, jednostavnost, potpora razmernom rastu broja čvorova, privatnost i nezavisnost.

Jedinstvenost logičkog imena podrazumijeva nepostojanje dvaju ili više logičkih čvorova s istim logičkim imenima unutar logičkog komunikacijskog prostora prividne

mreže. Ako je unutar jednog fizičkog komunikacijskog prostora uspostavljen veći broj prividnih komunikacijskih prostora, imena logičkih čvorova mogu se ponavljati sve dok pripadaju različitim prividnim komunikacijskim prostorima.

Jednostavnost logičkog imena zahtjeva uporabu logičkih imena koja su jednostavna za pamćenje od strane čovjeka kako bi upravljanje cijelokupnim prividnim komunikacijskim prostorom bilo što jednostavnije. Svojstvo jednostavnosti logičkog imena oprečno je svojstvu njegove jedinstvenosti. Jednostavna logička imena uglavnom podrazumijevaju nazive koje je moguće opisati kratkim tekstualnim podacima. S druge strane, raspodijeljeni programski sustavi izgrađeni nad prividnom mrežom često zahtijevaju sustav naslovljavanja posebno prilagođen području primjene te programsko dodjeljivanje logičkih imena pri dinamičkom dodavanju novih logičkih čvorova u sustav prividne mreže. Rješenje kojemu se u takvim slučajevima obično pribjegava je uporaba tablice preslikavanja jednostavnih, čovjeku razumljivih imena u imena pogodnija za programsku uporabu.

Sustav naslovljavanja logičkih čvorova treba omogućiti potporu razmernom rastu sustava obzirom na broj čvorova prividne mreže. Prividna mreža ima svojstvo razmernog rasta ako porast broja čvorova nema značajnog utjecaja na učinkovitost rada prividne mreže. Sustav naslovljavanja logičkih čvorova ima značajan utjecaj na učinkovitost usmjeravanja poruka među čvorovima pa njegovu osmišljavanju prethodi iscrpna analiza zahtjeva raspodijeljenog programskog sustava za koji se gradi prividna mreža. Na osnovi analize zahtjeva donosi se odluka o uporabi jednostavnijeg jednorazinskog ili složenijeg hijerarhijskog sustava imena. Jednorazinski sustav naslovljavanja je jednostavniji za programsko ostvarenje i učinkovitiji za prividne mreže s relativno malim brojem logičkih čvorova. Higerarhijski sustav naslovljavanja iziskuje složeniju programsku izvedbu, a najbolja svojstva pokazuje u prividnim mrežama s velikim brojem logičkih čvorova.

Svojstvom privatnosti nalaže se uporaba imena logičkog čvora koje skriva sve oblike povjerljivih informacija o funkciji, položaju, vlasniku ili sličnim svojstvima logičkog čvora. Naslovljavanje logičkih čvorova treba biti neovisno o položaju logičkog čvora unutar fizičke mreže te prijenosnom protokolu koji se na razini fizičke mreže koristi za prijenos podataka. Uporabom sustava naslovljavanja sa svojstvom nezavisnosti imena logičkih čvorova, prividna mreža ima svojstva pokretljivosti i neovisnosti o položaju unutar fizičke mreže, kao i tehnologiji prijenosa podataka unutar fizičkog komunikacijskog prostora.

3.1.3 Pretraživanje i usmjeravanje sadržaja

Usmjeravanje sadržaja naziv je za postupak prosljedivanja podatkovnih poruka od izvorišnog do odredišnog logičkog čvora. Usmjeravanje poruka u komunikacijskom prostoru prividne mreže obavljaju logički čvorovi. Ovisno o funkciji koju u određenom trenutku obavlja, logički čvor u komunikacijskom postupku može poprimiti tri različite uloge. Ako je logički čvor izvorište poruke, onda on poprima ulogu pošiljatelja poruke. Ako je logički čvor krajnje odredište poruke, on poprima ulogu primatelja poruke. Ako pošiljatelj i primatelj nisu povezani izravnom prividnom komunikacijskom vezom, slanje poruke obavlja se prosljedivanjem poruke putem više logičkih čvorova koji se nalaze na komunikacijskom putu između pošiljatelja i krajnjeg primatelja. Svi čvorovi koji prosljedivanjem poruke posreduju u komunikaciji između pošiljatelja i primatelja poprimaju ulogu usmjernika poruke. Zadatak usmjernika poruke je prosljedivanje poruke zaprimljene na ulaznoj komunikacijskoj vezi na jednu od izlaznih veza kako bi se nastavio njezin put do čvora primatelja. Odabir prividne komunikacijske veze kojom se proslijeđuje zaprimljena poruka naziva se postupkom usmjeravanja.

S postupcima usmjeravanja sadržaja povezani su postupci njegova pretraživanja. U podatkovno usmjerenoj prividnoj mreži pristup sadržaju ne ostvaruje se na osnovi adrese čvora koji raspolaže traženim sadržajem, već se njegovo pribavljanje obavlja postavljanjem upita. Postupci usmjeravanja sadržaja u takvim mrežama omogućuju učinkovito prosljedivanje postavljenih upita do onih čvorova mreže za koje postoji najveća vjerojatnost raspolaganja traženim sadržajem.

Ovisno o vrsti podataka koji čine osnovu za provedbu postupka usmjeravanja i pretraživanja sadržaja, prividne logičke mreže dijele se u tri skupine: mreže s usmjeravanjem poruka zasnovanim na imenima logičkih čvorova (engl. *node-centric networks*), mreže s usmjeravanjem poruka zasnovanim na sadržaju poruka (engl. *data-centric networks*) i mreže s usmjeravanjem poruka zasnovanim na prostornom rasporedu logičkih čvorova (engl. *position-centric networks*) [26].

Usmjeravanje poruka zasnovano na imenima logičkih čvorova

Tradicionalni način ostvarivanja postupaka usmjeravanja poruka u računalnim mrežama je usmjeravanje zasnovano na imenima ili adresama čvorova. Taj je pristup usmjeravanju poruka prvobitno bio iskorišten u poštanskom sustavu, gdje se usmjeravanje poštanskih pošiljaka obavlja na osnovi adrese primatelja na kojeg je pošiljka naslovljena. Isto načelo usmjeravanja poruka primjenjeno je u telekomunikacijskim mrežama za

povezivanje telefonskog broja s kojeg je upućen poziv i broja kojemu se taj poziv upućuje. Globalna računalna mreža Internet preuzeala je sustav usmjeravanja poruka zasnovan na adresama logičkih čvorova kao osnovicu mrežnog *IP* protokola. Usmjeravanje poruka u *IP* protokolu zasniva se na mrežnim adresama čvorova uslužne mreže, odnosno *IP* adresama.

Prividne logičke mreže s usmjeravanjem poruka zasnovanim na imenima logičkih čvorova pogodne su za izgradnju raspodijeljenih programskih sustava u kojima postoji jasna podjela funkcionalnosti usluge po čvorovima obrade. U takvim je programskim sustavima unaprijed poznato koje su funkcionalnosti i koji podaci dostupni na određenim logičkim čvorovima. U slučaju pokretljivih logičkih čvorova, bilo da se radi o pokretnom fizičkom čvoru ili o preseljenju logičkog čvora s jednog fizičkog čvora na drugi, funkcionalnosti i podaci sele se zajedno s logičkim čvorovima. Ako se logički čvor preseli s jednog fizičkog čvora na drugi, prividna mreža prikriva od raspodijeljenog programskog sustava promjene nastale na razini fizičke mreže. Funkcionalnostima i podacima koji su raspodijeljeni unutar logičkog komunikacijskog prostora prividne mreže raspodijeljeni programski sustav pristupa naslovljavanjem logičkih čvorova na kojima su oni dostupni.

Usmjeravanje poruka zasnovano na imenima logičkih čvorova provodi se održavanjem tablica usmjeravanja. Tablica usmjeravanja raspodijeljena je po svim logičkim čvorovima prividne mreže. U slučaju primjeka poruke koja nije naslovljena na njega, svaki logički čvor u svojoj tablici usmjeravanja sadrži informacije o tome kojem od logičkih čvorova s kojim ima uspostavljenu izravnu prividnu komunikacijsku vezu treba proslijediti primljenu poruku.

Algoritmi i postupci usmjeravanja zasnovanog na imenima logičkih čvorova brinu se za održavanje i ažuriranje raspodijeljene tablice usmjeravanja. Ažuriranje tablice usmjeravanja potrebno je provesti u slučajevima ulaska novog logičkog čvora u mrežu, u slučaju napuštanja mreže te u slučaju preseljenja logičkog čvora na drugi fizički čvor. Za prividne mreže s velikim brojem logičkih čvorova i puno prividnih komunikacijskih veza, tablica usmjeravanja može sadržavati veliki broj zapisa. Veličina tablice usmjeravanja ima nepovoljni utjecaj na količinu potrebnog spremničkog prostora za spremanje tablice te na brzinu pronalaska zapisa za usmjeravanje poruke. Utjecaj veličine tablice usmjeravanja naročito je izražen u slučajevima potrebe za izmjenama zapisa u tablici jer je lokalne promjene potrebno dojaviti i odgovarajućem broju udaljenih logičkih čvorova kako bi oni ažurirali vlastite tablice. Usmjeravanje zasnovano na imenima logičkih čvorova je, stoga, primjenjivo u prividnim mrežama u kojima se broj logičkih čvorova te oblik i ustroj mreže za vrijeme rada sustava rijetko mijenja.

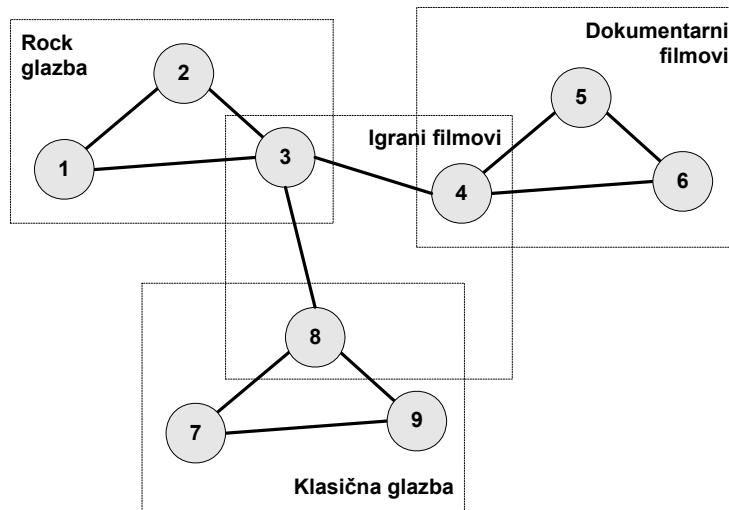
Usmjeravanje poruka zasnovano na sadržaju poruka

Prividne mreže s usmjeravanjem poruka zasnovanim na sadržaju poruka pogodne su za izgradnju raspodijeljenih programskih sustava u kojima su u središtu zanimanja podaci i osnovne operacije nad podacima, kao što su dohvati i spremanje podataka na udaljeno računalo. U takvim se programskim sustavima raspodijeljivanje funkcionalnosti i podataka obavlja u ovisnosti o informacijama sadržanim u podacima, a ne o čvorovima po kojima su ti podaci razmješteni. Algoritmi i postupci usmjeravanja poruka u takvim su mrežama nezavisni o broju logičkih čvorova i njihovim logičkim imenima. Logički čvor koji obavlja ulogu usmjernika donosi odluku o odabiru čvora kojemu treba prosljediti primljenu poruku na osnovi ispitivanja sadržaja poruke. Pošiljatelj poruke u postupku slanja poruke ne navodi ime logičkog čvora koji predstavlja krajnje odredište, već se poruka, ovisno o sadržaju koji nosi, samostalno usmjerava do odredišta.

Prividne mreže s usmjeravanjem poruka zasnovanim na sadržaju poruka izrazito su korisne za izgradnju podatkovno usmjerjenih raspodijeljenih programskih sustava sa svojstvom brzih dinamičkih promjena broja logičkih čvorova po kojima su raspodijeljene funkcionalnosti sustava te njihova međusobnog rasporeda. Za razliku od raspodijeljenog programskog sustava usmjerjenog logičkim čvorovima, gdje se upiti postavljaju u obliku „*Obavi funkcionalnost XYZ na logičkom čvoru ABC*”, podatkovno usmjereni raspodijeljeni programski sustavi postavljaju upite u obliku „*Dohvati podatak xyz*”, „*Spremi dokument abc.xml*” ili „*Dohvati crtane filmove iz 2003. godine i novije*”, bez navođenja imena čvora kojemu je upit upućen. Ovisno o izvedbi algoritma pretraživanja sadržaja, odgovor na upit može poslati prvi pronađeni logički čvor koji raspolaže traženim podacima ili svi čvorovi koji raspolažu tim podacima.

Primjer prividne mreže s usmjeravanjem poruka zasnovanim na sadržaju poruka su semantičke prividne mreže (engl. *Semantic Overlay Networks*) [32]. Na slici 3.5 prikazan je raspodijeljeni programski sustav za razmjenu multimedijskih sadržaja u obliku glazbe i filmova izgrađen u logičkom komunikacijskom prostoru semantičke prividne mreže. Budući da se radi o prividnoj mreži čije čvorove predstavljaju korisnička računala koja sudjeluju u razmjeni multimedijskih sadržaja, čvorovi se dinamično priključuju i napuštaju mrežu. Radi što učinkovitijeg upravljanja oblikom i ustrojem prividne mreže te provedbe postupka usmjeravanja poruka, logički se čvor u postupku pridruživanja mreži prividnim komunikacijskim vezama povezuje s onim logičkim čvorovima s kojima dijeli najviše semantičkih obilježja. Stvaranjem prividne mreže oblikuju se skupine čvorova semantički sličnog sadržaja. Postupak usmjeravanja poruka ispituje semantička obilježja njihova

sadržaja te ih usmjerava u onaj dio mreže u kojem postoji najveća vjerojatnost pronalaska traženih podataka.



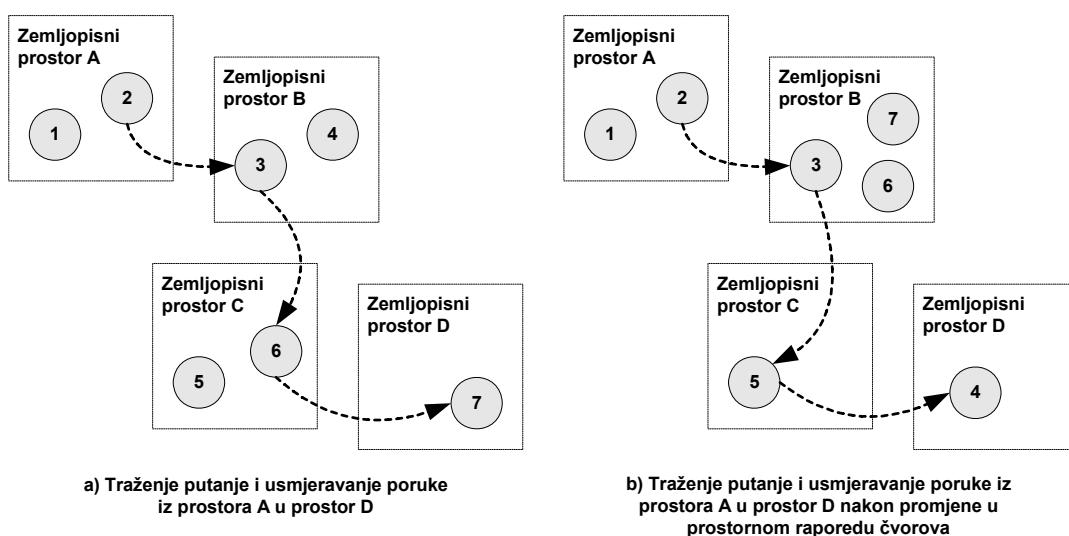
Slika 3.5 Raspodijeljeni programski sustav izgrađen u komunikacijskom prostoru semantičke prividne mreže

Usmjeravanje poruka zasnovano na prostornom rasporedu logičkih čvorova

Sveprisutno (engl. *ubiquitous computing*), prožimajuće (engl. *pervasive computing*) i neprimjetno računarstvo (engl. *invisible computing*) [33, 34, 35] predstavljaju nove oblike primjene računarskih znanosti i tehnologija u svakodnevnom životu pojedinca i zajednice. Iako tek u povojima te su grane računarstva utvrdile nekoliko jasnih smjernica za oblikovanje budućih primjenskih usluga na tom području. Sustavi neprimjetne inteligencije (engl. *intelligent computer systems*) u obliku osjetila (engl. *sensor*) i pokretača (engl. *actuator*) ugrađenih u okolinu u kojoj čovjek živi i djeluje trebaju u što većoj mjeri biti svjesni stanja okoline u koju su postavljeni. Doношење odluke o povratnom djelovanju na okolinu zasniva se na promatranju stanja i prikupljanju podataka iz vlastite okoline te suradnji i razmjeni podataka sa sličnim sustavima iz iste ili drugih okolina. Bitan čimbenik u radu sustava neprimjetne inteligencije jest poznavanje prostornog razmještaja osjetila i pokretača unutar okoline te određivanje položaja s kojeg pristižu informacije, odnosno na koji treba uputiti poruku o djelovanju na okolinu. Stoga, jedno od važnih područja primjene sveprisutnog, prožimajućeg i neprimjetnog računarstva pripada uslugama svjesnih vlastitog prostornog položaja (engl. *location-based services*) [36].

Usluge svjesne prostornog položaja zahtijevaju poseban pristup ostvarivanju komunikacije među raspodijeljenim mrežnim čvorovima u obliku osjetila, pokretača i nadzornih uređaja. Za takve je usluge bitno informaciju pribaviti ili je uputiti na točno

određeno mjesto u prostoru, dok je manje važno koji od raspodijeljenih mrežnih čvorova tu informaciju prihvata i obrađuje te od kojeg je čvora informacija preuzeta. Poteškoće koje prate ostvarivanje komunikacije u sustavima neprimjetne inteligencije uzrokovane su pokretljivošću čvorova i njihovom povremenom nedostupnošću. Sustavi neprimjetne inteligencije se, naime, ostvaruju kao bežične mreže osjetila i pokretača koji izvor napajanja crpe iz baterijskih spremnika energije. Kako bi se očuvao vijek trajanja baterijskog izvora napajanja, uređaji se automatski povremeno isključuju, a njihova povremena nedostupnost uzrokovana je i pražnjenjem baterijskog izvora.



Slika 3.6 Usmjeravanje poruka zasnovano na položaju logičkih čvorova

Za ostvarivanje komunikacije među čvorovima sustava neprimjetne inteligencije važno je usmjeravanje poruka zasnovano na prostornom rasporedu čvorova. Tijekom upućivanja poruke s jednog logičkog čvora na drugi, čvor pošiljatelj ne navodi ime čvora primatelja, nego prostorni položaj odakle nastoji pribaviti podatke. Ako unutar zadanih prostora postoji čvor koji raspolaze traženim podacima, on poslužuje pošiljateljev zahtjev. Slikom 3.6 prikazan je jednostavni programski sustav koji koristi usmjeravanje poruke zasnovano na prostornom rasporedu čvorova. Okolina sustava podijeljena je na četiri zemljopisno udaljena prostora. Na slici 3.6.a prikazan je put poruke upućene od čvora iz prostora A u prostor D. Usmjeravanjem poruke putem čvorova 3 i 6, poruka stiže do čvora 7 u prostoru D koji je obrađuje. Slikom 3.6.b prikazan je isti slučaj s vremenskim odmakom u kojem je došlo do promjene prostornog rasporeda čvorova. Čvor 6 preselio je iz prostora C u prostor B, dok su čvorovi 4 i 7 zamijenili svoja mjesta u prostorima B i D. Za postizanje istog cilja kao i u prethodnom slučaju, ponovno je potrebno uputiti poruku iz prostora A u

prostor D. Međutim, u skladu s novonastalim rasporedom čvorova, put poruke sada vodi preko čvorova 3 i 5, a poruku obrađuje čvor 4. Usmjeravanjem poruke zasnovanim na prostornom rasporedu čvorova omogućeno je putovanje dviju potpuno jednakih poruka s istim krajnjim odredištem uz posredovanje različitih čvorova, dok je krajnja obrada poruke također obavljena od strane dvaju različitih čvorova.

3.1.4 Sigurnost prijenosa podataka

Sigurnost prijenosa podataka u prividnim logičkim mrežama ima jednako važnu ulogu kao i u svim suvremenim komunikacijskim sustavima. Osiguravanje sigurnog prijenosa podataka obuhvaća postupke za utvrđivanje izvornosti pristupnog čvora (engl. *authentication*), provjeru njegovih prava pristupa do tražene funkcionalnosti ili podataka (engl. *authorization, access control*), provjeru vjerodostojnosti i cjelovitosti primljenih poruka (engl. *integrity*), očuvanje privatnosti mrežnih čvorova i korisnika (engl. *privacy, anonymity*) te očuvanje tajnosti sadržaja poruka (engl. *confidentiality, secrecy*).

Tablica 3.1 Tablica prava pristupa do funkcionalnosti i podataka dostupnih na lokalnom logičkom čvoru

		Operacije		
		Čitanje	Pisanje	Pozivanje
S r e d s t v a	Dokument <i>pritisak.xml</i>	čvor 1, čvor 2, čvor 3	čvor 1	-
	Dokument <i>temperatura.xml</i>	čvor 1, čvor 2, čvor 3	čvor 2	-
	Funkcija <i>MjeriPritisakZraka</i>	-	-	čvor 1
	Funkcija <i>MjeriTemperaturuZraka</i>	-	-	čvor 2

Utvrđivanje izvornosti pristupnih čvorova, utvrđivanje vjerodostojnosti i cjelovitosti poruka te očuvanje tajnosti njihova sadržaja provodi se primjenom postupaka kriptiranja i digitalnog potpisivanja podataka. Kriptiranje i digitalno potpisivanje su kriptografskim metodama i algoritmima poduprti zaštitni postupci protiv zlonamjernog iskorištavanja mrežnih sredstava. Primjenom kriptografije tajnih ključeva osigurava se očuvanje tajnosti sadržaja poruka, dok se primjenom kriptografije javnih ključeva osigurava razmjena tajnih ključeva, digitalno potpisivanje podataka te utvrđivanje izvornosti pristupnih čvorova [37]. Provjera prava pristupa do raspodijeljenih funkcionalnosti i podataka provodi se definiranjem pristupnih tablica (engl. *access control list*) [38] u kojima se za svaku funkcionalnost i skupinu podataka navode pristupna prava svakog od pristupnih čvorova. Pristupna tablica je dvodimenzionalna struktura u kojoj jednu dimenziju čine štićena sredstva, odnosno funkcionalnosti i podaci do kojih se štiti pristup, a drugom su dimenzijom

definirane dozvoljene operacije nad tim sredstvima. Na presjecištu dviju dimenzija popisani su mrežni čvorovi koji imaju pravo obaviti određenu operaciju nad određenim sredstvom. Tablicom 3.1 prikazan je primjer pristupne tablice za čvor prividne mreže koji obavlja funkciju osjetila za mjerjenje temperature i pritiska zraka. Pretpostavka je da se, osim čvora koji obavlja funkciju osjetila, prividna mreža sastoji od još tri logička čvora naslovljena logičkim imenima *čvor 1*, *čvor 2* i *čvor 3*.

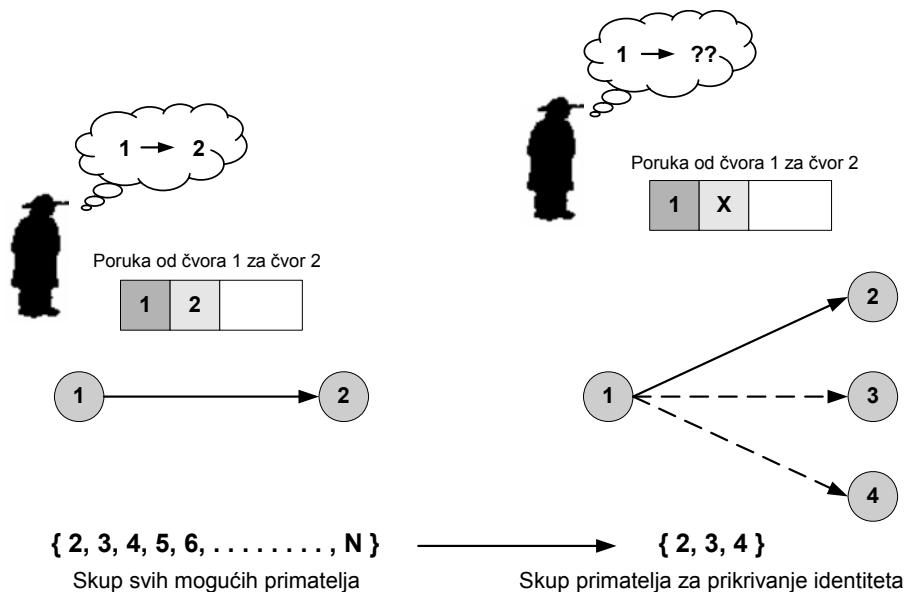
Funkcionalnost koja je za suvremene raspoložive programske sustave, a naročito za sustave neprimjetne inteligencije od izuzetnog značaja jest prikrivanje identiteta sudionika komunikacije i očuvanje njihove privatnosti. Budući da je riječ o sustavima koji su ugrađeni u čovjekovu svakodnevnu okolinu, način njihova djelovanja treba uključivati i sociološku dimenziju kojom ih ljudi doživljavaju. Sustav koji bi načinom svojeg djelovanja u svakom trenutku otkriva gdje se određeni pojedinac nalazi i što radi, narušavajući na taj način privatnost korisnika, zasigurno ne bi naišao na širu prihvaćenost od strane društvene zajednice. Narušavanje privatnosti korisnika raspoloživenih programskih sustava ugroženo je na dva osnovna načina. Uvidom u sadržaj poruka koji sudionici međusobno razmijenjuju dolazi se u posjed povjerljivih korisničkih informacija. Uvidom u identitet korisnika koji sudjeluju u komunikaciji narušena je privatnost korisničkih međusobnih odnosa.

Sustavima za siguran prijenos podataka u postojećim fizičkim mrežama pridavala se značajna pažnja te su oni u postojećim komunikacijskim sustavima široko primjenjivani. Međutim, iako postojeći sigurnosni sustavi dobro rješavaju problem zaštite tajnosti sadržaja prenesenih poruka, razvijeno je vrlo malo sigurnosnih sustava koji štite privatnost i identitet sudionika mrežne komunikacije. Većina komunikacijskih sustava u širokoj primjeni, uključujući i globalnu mrežu Internet, koristi informacije o pošiljatelju i primatelju u postupku usmjeravanja poruka kroz mrežu čvorova. Čvorovi usmjernici održavaju tablice usmjeravanja te usmjeravanje poruke obavljaju uspoređivanjem primljenih informacija o pošiljatelju i primatelju poruke i informacija o čvoru na koji treba usmjeriti poruku. Informacije o primatelju i pošiljatelju poruke sadržane su u samoj poruci, dok se informacije o čvoru na koji treba usmjeriti poruku dobivaju pretraživanjem tablice usmjeravanja. Budući da su informacije o pošiljatelju i primatelju potrebne u postupku usmjeravanja poruka kroz mrežu, njihovim se skrivanjem onemogućava rad sustava usmjeravanja. Jedan od mogućih načina prikrivanja identiteta sudionika jest preoblikovanje sustava usmjeravanja. Informacije o usmjeravanju poruke kroz mrežu nisu sadržane u tablicama usmjeravanja čvorova usmjernika, već se uključuju u samu poruku.

Sigurnosni mehanizam koji osigurava tajnost prenesenih podataka, a ujedno i prikriva identitet sudionika mrežne komunikacije zahtjeva značajne izmjene u načinu rada sustava usmjeravanja poruka. Tijekom dosadašnjeg razvoja sigurnosnih sustava za prijenos podataka bez izravnog imenovanja sudionika mrežne komunikacije osmišljeno je nekoliko rješenja. U nastavku ovog poglavlja ukratko su opisana načela rada dvaju takvih sustava, sustava s maskiranjem imena i preplavljanjem i sustava ugnježdenih ljudsaka. Budući da oba sustava zahtijevaju posebno prilagođen sustav usmjeravanja poruka, za njihovu primjenu u mreži Internet potrebno je izgraditi odgovarajuće prividne mreže s prilagođenim postupcima usmjeravanja poruka.

Sustav s maskiranjem imena i preplavljanjem

Najjednostavniji način prikrivanja identiteta sudionika koji sudjeluju u komunikaciji računalnom mrežom jest korištenje zamjenskih imena umjesto stvarnih imena sudionika te istovremenim upućivanjem poruke većem broju sudionika, umjesto primatelju samom. Sigurnosni sustav koji koristi takvo načelo rada naziva se sustavom s maskiranjem imena i preplavljanjem (engl. *implicit addressing and broadcasting*) [39].



Slika 3.7 Sigurnosni sustav za prikrivanje identiteta sudionika mrežne komunikacije s maskiranjem imena i preplavljanjem

Načelo rada sustava s maskiranjem imena i preplavljanjem prikazano je slikom 3.7. Ako logički čvor 1 nastoji poslati poruku logičkom čvoru 2, on uz čvor 2 iz skupa svih mogućih primatelja slučajnim odabirom odabire još nekoliko lažnih primatelja, primjerice 3 i

4. Stvarni primatelj poruke zajedno sa svim lažnim primateljima predstavlja skup primatelja za prikrivanje identiteta. Tijekom slanja pošiljatelj upućuje poruku svim članovima skupa za prikrivanje identiteta. Umjesto njegovim stvarnim imenom, pošiljatelj primatelja poruke naslovljava lažnim imenom, primjerice imenom X. Lažno ime X je posebna vrsta informacije koju dijele čvorovi 1 i 2, odnosno pošiljatelj poruke i njezin stvarni primatelj. Značenje imena X nepoznato je svim ostalim sudionicima. Stvarni primatelj po primitku poruke naslovljene imenom X utvrđuje da je to zaštićena poruka namjenjena upravo njemu. Preostali čvorovi iz skupa za prikrivanje identiteta nisu u mogućnosti obraditi poruku zbog nepoznavanja značenja lažnog imena te je odbacuju kao nevažeću.

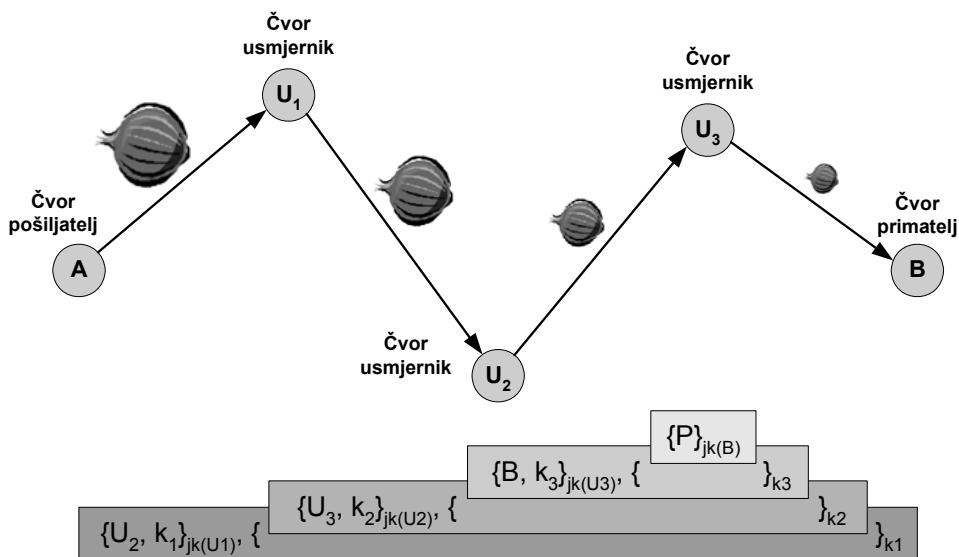
Sustav s maskiranjem imena i preplavljanjem pruža zadovoljavajuću zaštitu identiteta sudionika pod pretpostavkom da nijedan od sudionika iz skupa za prikrivanje identiteta nije uljez u komunikacijskom kanalu. Ako su u primjeru prikazanom na slici 3.7 sudionici 3 i 4 uljezi, tada je moguće jednostavno otkriti identitet stvarnog primatelja poruke. Drugi nedostatak sustava s maskiranjem imena i preplavljanjem očituje se u preplavljanju mreže lažnim porukama za zbumjivanje napadača. U mrežama s velikim brojem čvorova preplavljanje može značajno utjecati na učinkovitost komunikacijskog sustava s gledišta brzine prijenosa podataka, količine mrežnog prometa te mogućnosti razmjernog rasta mreže.

Sustav ugnježdenih ljsaka

Dva osnovna nedostatka sustava s maskiranjem imena i preplavljanjem, potreba za međusobnim povjerenjem svih sudionika iz skupa za prikrivanje identiteta i velika vjerojatnost zagušenja mreže u slučaju velikog broja čvorova, riješena su u sustavu za prikrivanje identiteta korištenjem ugnježdenih ljsaka (engl. *onion routing*) [40]. Sustav ugnježdenih ljsaka razvila je američka vojna mornarica sredinom devedesetih godina dvadesetog stoljeća.

Načelo rada sustava ugnježdenih ljsaka prikazano je slikom 3.8. Prikrivanje identiteta sudionika mrežne komunikacije zasniva se na neizravnoj komunikaciji između pošiljatelja i primatelja poruke te primjenom kriptografskih postupaka za skrivanje sadržaja poruka. Poruke se prije isporuke primateljima usmjeravaju kroz određeni broj čvorova usmjernika. Pošiljatelj poruke oblikuje poruku za primatelja i kriptira je njegovim javnim ključem. Na taj način je zaštićena tajnost sadržaja poruke. Nakon toga pošiljatelj odabire skup čvorova putem kojih poruku usmjerava do određenog čvora. Poruci kriptiranoj javnim ključem primatelja pošiljatelj dodaje ime posljednjeg čvora u lancu čvorova usmjernika te ukupnu

poruku kriptira javnim ključem posljednjeg usmjernika u lancu. Nakon toga tako oblikovanoj poruci dodaje ime pretposljednjeg usmjernika u lancu i kriptira sveukupnu poruku njegovim javnim ključem. Postupak se ponavlja sve dok se ne dobije kriptirana poruka koja sadrži ime prvog usmjernika u lancu. Tako oblikovana poruka šalje se prvom usmjerniku u lancu. Prvi usmjernik dekriptira sadržaj poruke svojim tajnim ključem, doznaće ime sljedećeg usmjernika u lancu i prosljeđuje mu primljenu poruku. Kada poruka konačno pristigne na odredište, primatelj poruke je dekriptira svojim tajnim ključem i doznaće njezin izvorni sadržaj.



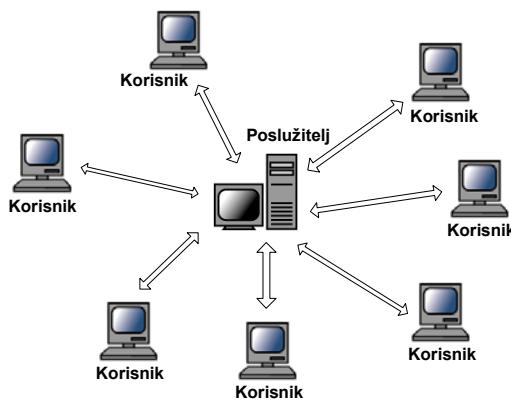
Slika 3.8 Sigurnosni sustav za prikrivanje identiteta sudionika mrežne komunikacije s ugnježdenim ljudskama

Svaki čvor u komunikacijskom lancu, osim pošiljatelja i primatelja, doznaće samo ime sljedećeg usmjernika u lancu, dok mu sve ostale informacije o pošiljatelju, primatelju te putu poruke prije i nakon njega ostaju skrivene. Čvor posrednik ne može zaključiti da li je čvor od kojeg je poruka pristigla pošiljatelj ili prethodni usmjernik u lancu. Također, čvor posrednik ne zna da li je čvor kojemu prosljeđuje poruku sljedeći usmjernik u lancu ili njezin krajnji primatelj. Sigurnosni sustav ugnježdenih ljudsaka, osim što štiti identitet sudionika mrežne komunikacije, čuva i tajnost sadržaja prenošenih poruka. Nedostatak sustava ugnježdenih ljudsaka je određivanje puta poruke kroz mrežu čvorova usmjernika na izvorištu poruke. U slučaju kvara jednog od čvorova usmjernika, sustav nije u mogućnosti provesti postupak oporavka od pogreške koji odabire zamjenski put poruke do odredišta.

3.2 Mreže ravnopravnih sudionika

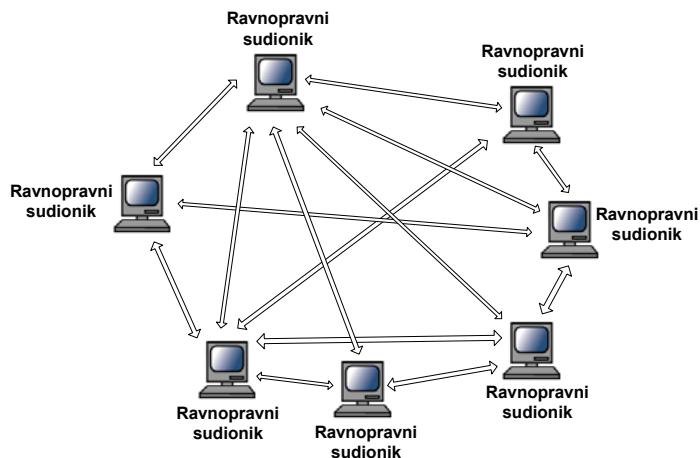
Mreže ravnopravnih sudionika (engl. *peer-to-peer networks*) [44] jedno su od nekoliko velikih područja primjene prividnih logičkih mreža. Namjena mreža ravnopravnih sudionika je dijeljenje računalnih sredstava, primjerice računalne snage i spremničkog prostora, mrežnih sredstava, kao što su komunikacijske veze i širina prijenosnog opsega te funkcionalnosti i podataka primjenske razine među svim čvorovima koji čine računalnu mrežu. Mreže ravnopravnih sudionika u funkcionalnom i organizacijskom smislu predstavljaju oprečni pristup raspodijeljenim programskim sustavima zasnovanim na korisničko-poslužiteljskoj arhitekturi (engl. *client-server architecture*) [6, 7].

Većina raspodijeljenih programskih sustava globane računalne mreže Internet u današnje vrijeme koristi korisničko-poslužiteljsku arhitekturu za ostvarivanje i povezivanje raspodijeljenih programskih funkcionalnosti. Korisničko-poslužiteljska arhitektura sastoji se od jednog mrežnog elementa visoke učinkovitosti i više elemenata manje učinkovitosti. Element visoke učinkovitosti naziva se poslužiteljem (engl. *server*), dok se elementi manjih učinkovitosti nazivaju korisnicima (engl. *client*). Poslužitelj je element programskog sustava na kojem su postavljene gotovo sve funkcionalnosti i svi podaci korišteni u postupku pružanja usluge. Funkcionalnosti i podaci smješteni na poslužitelju zajednički su svim korisnicima u sustavu. Funkcionalnosti i podaci korisničkih elemenata koriste se samo pri lokalnoj obradi. Korisnici pozivaju funkcionalnosti ili koriste podatke dostupne na poslužitelju te uporabom zajedničkih poslužiteljskih i vlastitih lokalnih funkcionalnosti i podataka postiću željenu korisničku funkcionalnost. Primjer raspodijeljenog programskog sustava zasnovanog na korisničko-poslužiteljskoj arhitekturi prikazan je slikom 3.9.



Slika 3.9 Raspodijeljeni programski sustav zasnovan na korisničko-poslužiteljskoj arhitekturi

Korisničko-poslužiteljska arhitektura pogodna je za razvoj raspodijeljenih programskih sustava u kojima postoji jasno razdvajanje funkcionalnosti i podataka na poslužiteljske i korisničke elemente. Raspodijeljeni programski sustavi koji zahtijevaju ravnopravnu raspodjelu funkcionalnosti i podataka po svim mrežnim čvorovima iziskuju i prilagodbu mrežne arhitekture. Izjednačavanje učinkovitosti mrežnih elemenata postiže se arhitekturom mreže ravnopravnih sudionika. U mreži ravnopravnih sudionika ponašajna svojstva svih članova istovremeno poprimaju obilježja poslužitelja i korisnika. Funkcionalnosti i podaci raspodijeljenih programskih sustava podjednako su raspoređene po svim članovima mreže ravnopravnih sudionika. Raspodijeljeni programski sustav zasnovan na arhitekturi mreže ravnopravnih sudionika prikazan je slikom 3.10.



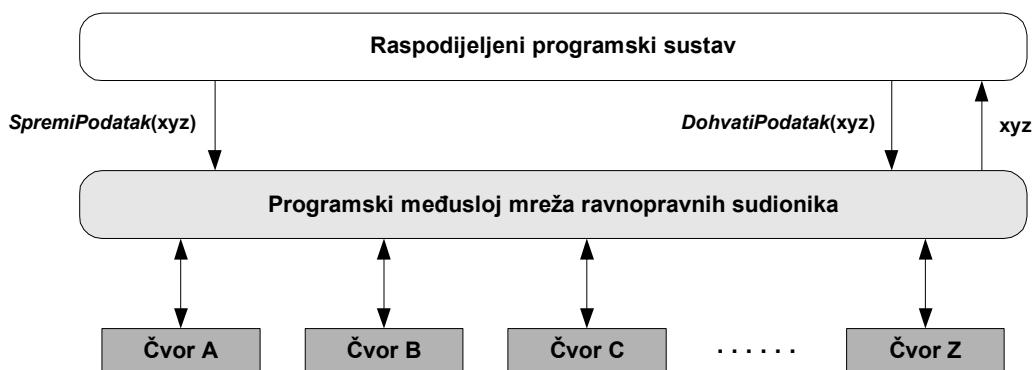
Slika 3.10 Raspodijeljeni programski sustav zasnovan na mreži ravnopravnih sudionika

Računalni sustav zasnovan na mreži ravnopravnih sudionika ima svojstvo dinamičnosti, što znači da sudionici mogu nepredviđeno pristupati i napuštati mrežu. Stoga je jedno od bitnih svojstava mreže ravnopravnih sudionika zalihost raspodijeljenih funkcionalnosti i podataka. Raspodjeljivanje funkcionalnosti i podataka ostvareno je raspodjeljivanjem istih funkcionalnosti i podataka na više čvorova mreže. U slučaju napuštanja mreže, funkcionalnosti i podaci dostupni na čvoru koji je napušta ne smiju biti u potpunosti nedostupni.

3.2.1 Arhitektura mreže ravnopravnih sudionika

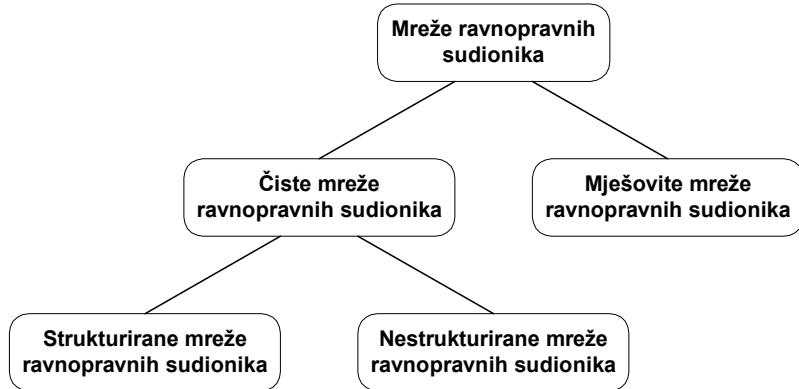
Ostvarivanje mreža ravnopravnih sudionika na postojećoj komunikacijskoj infrastrukturi mreže Internet postiže se izgradnjom prividnih mreža. Prividne mreže grade se u obliku programskih komponenata koje osiguravaju stvaranje programskog međusloja (engl. *peer-to-peer middleware*) [8] iznad fizičke komunikacijske mreže. Programski

međusloj mreža ravnopravnih sudionika postavljen je na svakom čvoru koji čini mrežu ravnopravnih sudionika. Uloga programskog međusloja je razdvajanje fizičke komunikacijske mreže od primjenske logike raspodijeljenog programskog sustava, odnosno pretvorba fizičke mreže u prividnu mrežu prilagođenu potrebama raspodijeljenog programskog sustava. Raspodijeljeni programski sustavi koji koriste mrežu ravnopravnih sudionika kao prostor za komunikaciju i raspodjelu programske funkcionalnosti grade se iskorištanjem funkcionalnosti programskog međusloja. Najvažnije funkcionalnosti koje programski međusloj pruža raspodijeljenim programskim sustavima su pretraživanje sadržaja u obliku funkcionalnosti ili podataka te usmjeravanje poruka kojima se taj sadržaj prenosi. Slikom 3.11 prikazana je arhitektura raspodijeljenog programskog sustava zasnovanog na mreži ravnopravnih sudionika.



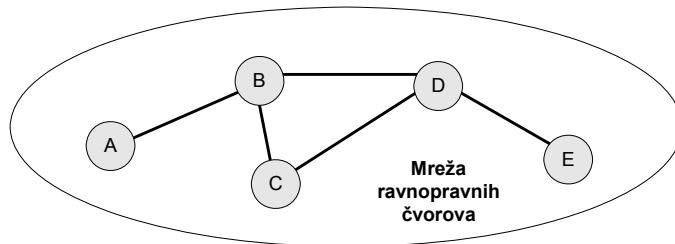
Slika 3.11 Arhitektura raspodijeljenog programskog sustava zasnovanog na mreži ravnopravnih sudionika

Obzirom na oblik i ustroj, mreže ravnopravnih sudionika svrstavaju se u dvije osnovne skupine, ovisno o tome da li ravnopravni čvorovi samostalno održavaju članstvo, oblik i ustroj mreže ili se u te svrhe koristi izdvojeni upravljački čvor. Ako su postupci upravljanja ustrojem mreže prepusteni ravnopravnim sudionicima koji sudjeluju u oblikovanju mreže, mreže se nazivaju čistim mrežama ravnopravnih sudionika (engl. *pure peer-to-peer networks*). Ako je za provedbu postupaka upravljanja ustrojem mreže zadužen izdvojeni upravljački čvor, mreže se nazivaju mješovitim mrežama ravnopravnih sudionika (engl. *hybrid peer-to-peer networks*). Čiste mreže ravnopravnih sudionika dijele se na strukturirane i nestrukturirane mreže, ovisno o načinu održavanja logičkog rasporeda čvorova i njihovih komunikacijskih veza. Način upravljanja ustrojem mreže ravnopravnih sudionika utječe na provedbu postupaka pretraživanja i usmjeravanja sadržaja među ravnopravnim čvorovima. Podjela mreža ravnopravnih sudionika prema načinu upravljanja ustrojem mreže te pretraživanjem i usmjeravanjem sadržaja prikazana je slikom 3.12.



Slika 3.12 Podjela mreža ravnopravnih sudionika prema načinu upravljanja ustrojem mreže te pretraživanjem i usmjeravanjem sadržaja

U čistoj mreži ravnopravnih sudionika svi su čvorovi u potpunosti ravnopravni. Arhitektura čiste mreže ravnopravnih sudionika prikazana je slikom 3.13. Logičke komunikacijske veze između čvorova uspostavljaju se na osnovi izravnog međusobnog povjerenja među čvorovima. Svaki čvor čiste mreže ravnopravnih sudionika je nezavisni mrežni element koji u svakom trenutku može napustiti mrežu ili se u nju učlaniti.



Slika 3.13 Arhitektura čiste mreže ravnopravnih sudionika

U strukturiranim mrežama ravnopravnih sudionika postoje stroga pravila za održavanje ustroja mreže i definiranje logičkih komunikacijskih veza među čvorovima. Održavanje ustroja mreže na osnovi unaprijed utvrđenih pravila omogućava izgradnju učinkovitih postupaka pretraživanja i usmjeravanja sadržaja, ali održavanje definirane strukture mreže zahtijeva provedbu složenih postupaka tijekom učlanjivanja čvorova u mrežu i napuštanja mreže. Primjeri strukturiranih mreža ravnopravnih sudionika u sustavi *Chord* [45], *P-Grid* [46], *Freenet* [47] i *OceanStore* [48]. Nestrukturirane mreže ne definiraju ograničenja za održavanje ustroja mreže. Logički raspored i povezanost čvorova nestrukturirane mreže ravnopravnih sudionika oblikuje se prema nahođenju čvorova sudionika. Učlanjivanje i napuštanje nestrukturiranih mreža provodi se jednostavnim i učinkovitim postupcima, ali je zbog nepostojanja pravila za održavanje logičkog rasporeda

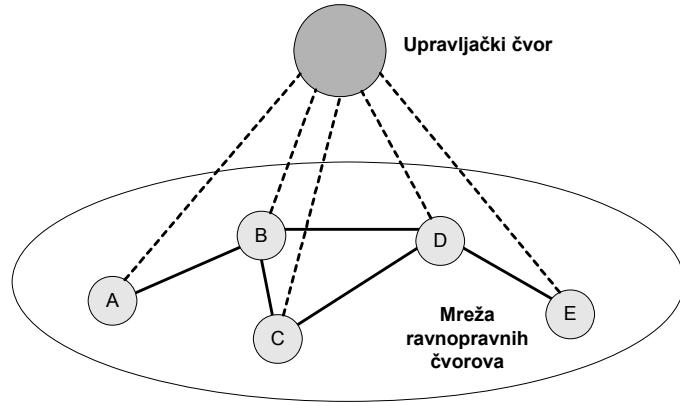
čvorova pretraživanje i usmjeravanja sadržaja znatno otežano. Primjer nestrukturirane mreže ravnopravnih sudionika je mreža *Gnutella* [49].

Tijekom učlanjivanja u mrežu čvor se povezuje sa skupom ostalih ravnopravnih čvorova koji već jesu sudionici mreže. U slučaju strukturirane mreže ravnopravnih sudionika čvor se povezuje sa skupom čvorova određenim pravilima o održavanju ustroja mreže. Kako bi se održala definirana struktura mreže, raspodjelu sadržaja i logičke veze među čvorovima u dijelu mreže u kojem se nalazi novoučlanjeni čvor potrebno je prilagoditi novonastalim okolnostima. U slučaju nestrukturirane mreže čvor se povezuje s proizvoljnim skupom čvorova. Prilagodba raspodjele sadržaja i logičkih veza među čvorovima u takvim mrežama nije potrebna. Tijekom napuštanja mreže čvor koji je napušta mora o tome obavijestiti čvorove mreže s kojima je povezan. Slično kao i u postupku učlanjivanja, tijekom izlaska čvora iz mreže u strukturiranim je mrežama potrebno provesti prilagodbu mreže novonastalim okolnostima radi očuvanja strukture mreže. U nestrukturiranim mrežama prilagodba nije potrebna.

Poruke između čvora pošiljatelja i čvora primatelja usmjeravaju se putem odgovarajućeg broja čvorova na putu između tih dvaju čvorova. U strukturiranim mrežama ravnopravnih sudionika usmjeravanje poruka zasnovano je na raspodijeljenoj tablici usmjeravanja. Svaki čvor mreže sadrži dio tablice usmjeravanja s podacima o usmjeravanju koji se odnose na dio mreže u kojem se nalazi. Na osnovi podataka u tablici usmjeravanja, čvor usmjerava poruku na jedan od čvorova s kojima je povezan i putem kojeg poruka najbrže stiže do odredišta. U nestrukturiranim mrežama ravnopravnih sudionika svaki čvor koji primi poruku za čiju obradu nije zadužen proslijeđuje tu poruku svim čvorovima s kojima je povezan. Put poruke od izvorišnog do odredišnog čvora pronalazi se postupkom preplavljanja mreže (engl. *flooding*). Nakon što primi poruku za čiju je obradu zadužen, čvor prekida postupak preplavljanja mreže. Za spriječavanje višestrukog proslijedivanja iste poruke i nastajanja zatvorenih petlji u mreži, poruke su obilježene posebnim oznakama, a čvorovi vode bilješke o proslijedenim porukama.

Druga skupina mreža ravnopravnih sudionika obzirom na način upravljanja oblikom i ustrojstvom mreže su mješovite mreže ravnopravnih sudionika. Mješovita mreža ravnopravnih sudionika je kombinacija čiste mreže ravnopravnih sudionika i korisničko-poslužiteljske arhitekture. Funkcionalnosti primjenske logike raspodijeljene su po ravnopravnim čvorovima. Upravljačke funkcionalnosti za upravljanje oblikom i ustrojem mreže, upravljanje članstvom u mreži te pretraživanje i usmjeravanje sadržaja objedinjene su

u izdvojenom upravljačkom čvoru. Arhitektura mješovite mreže ravnopravnih sudionika prikazana je slikom 3.14.



Slika 3.14 Arhitektura mješovite mreže ravnopravnih sudionika

Komunikacija između ravnopravnih sudionika uspostavlja se posredovanjem upravljačkog čvora. Čvor koji nastoji pristupiti do funkcionalnosti ili podataka raspoloživih na nekom od udaljenih čvorova mreže obraća se upravljačkom čvoru s ciljem pronalaska čvora koji raspolaze potrebnim sadržajem. Nakon što od upravljačkog čvora dohvati ime čvora s kojim treba uspostaviti vezu, dva čvora nastavljaju komunicirati izravno. Mreže ravnopravnih sudionika u svom prvobitnom obliku bile su upravo mješovite mreže ravnopravnih sudionika. Najpoznatiji primjer mješovite mreže ravnopravnih sudionika je mreža *Napster* [51] za razmjenu glazbe putem mreže Internet.

3.2.2 Programski međusloj mreža ravnopravnih sudionika

Ključna uloga za pravilan i učinkovit rad raspodijeljenog programskog sustava zasnovanog na mreži ravnopravnih sudionika pripada programskom međusloju mreže ravnopravnih sudionika. Programski međusloj mreže ravnopravnih sudionika je raspodijeljeni programski sustav koji preobražava postojeću fizičku komunikacijsku mrežu u prividnu mrežu prilagođenu području primjene raspodijeljenog programskog sustava. Funkcionalnosti koje programski međusloj mreže ravnopravnih sudionika izlaže na korištenje raspodijeljenom programskom sustavu uključuju upravljanje članstvom logičkih čvorova, upravljanje oblikom i ustrojem prividne mreže, pretraživanje i usmjeravanje poruka među članovima mreže, upravljanje pravima pristupa do raspodijeljenih funkcionalnosti i podataka te sigurnost prijenosa podataka među članovima mreže.

Dvije najvažnije funkcionalnosti programskog međusloja mreže ravnopravnih sudionika su upravljanje oblikom i ustrojem prividne mreže te pretraživanje i usmjeravanje

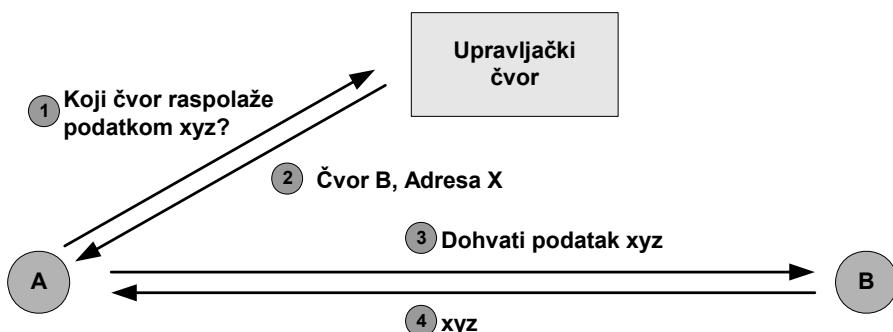
sadržaja u mreži ravnopravnih sudionika. Oblikovanje i izvedba ovih dviju funkcionalnosti određuje način izvedbe, razinu složenosti i učinkovitost svih ostalih funkcionalnosti. Upravljanje oblikom i ustrojem prividne mreže obuhvaća aktivnosti koje se provode za vrijeme učlanjivanja novog čvora u mrežu te za vrijeme napuštanja mreže od strane postojećeg čvora. Tijekom ulaska novog čvora u mrežu ravnopravnih sudionika, potrebno ga je povezati s postojećim čvorovima u mreži te prilagoditi postupak usmjeravanja kako bi se omogućilo slanje poruka s novog čvora prema ostalim čvorovima u mreži, kao i slanje poruka s ostatka mreže prema novoučlanjenom čvoru. Tijekom napuštanja mreže potrebno je provesti obrnuti postupak. Informacije o čvoru koji napušta mrežu moraju biti uklonjene iz tablica usmjeravanja, dok se preoblikovanjem oblika i ustroja mreže čuva njegina povezanost.

Način upravljanja oblikom i ustrojem mreže ravnopravnih sudionika te postupci pretraživanja i usmjeravanja sadržaja uvelike su ovisni o arhitekturi mreže. U mješovitim mrežama ravnopravnih sudionika provedba tih postupaka prepustena je upravljačkom čvoru. U čistim mrežama ravnopravnih sudionika postupci upravljanja mrežom i sadržajem raspodijeljeni su po svim ravnopravnim sudionicima mreže. Upravljanjem oblikom i ustrojem mreže te pretraživanje i usmjeravanje sadržaja u čistim mrežama ravnopravnih sudionika provodi se na dva osnovna načina. Jednostavniji, ali manje učinkovit način zasnovan je na preplavljivanju mreže (engl. *flooding*) upitima za pretraživanje sadržaja [49]. Taj se postupak koristi za pretraživanje i usmjeravanje sadržaja u nestrukturiranim mrežama ravnopravnih sudionika. Složeniji, ali ujedno i znatno učinkovitiji postupak zasniva se na korištenju funkcije sažimanja sadržaja (engl. *hash function*) i raspodijeljene tablice sažetaka (engl. *distributed hash table*) [45]. Funkcija sažimanja sadržaja primjenjuje se nad logičkim imenima čvorova sudionika i sadržajem kojim raspolažu. Sažeci imena logičkih čvorova i sadržaja koji nastaju kao rezultat primjene funkcije sažimanja sadržani su u raspodijeljenoj tablici sažetaka. Ovaj postupak pretraživanja i usmjeravanja sadržaja svojstven je strukturiranim mrežama ravnopravnih sudionika.

Upravljanje mrežom zasnovano na središnjem upravljačkom čvoru

Provedba postupaka upravljanja članstvom u mreži, oblikom i ustrojstvom mreže, pretraživanjem i usmjeravanjem sadržaja i sigurnošću komunikacije zasnovana na središnjem upravljačkom čvoru primjenjuje se u mješovitim mrežama ravnopravnih sudionika. Tijekom ulaska u mrežu čvor se prijavljuje pri upravljačkom čvoru. Tijekom postupka prijave novoučlanjeni čvor dojavljuje upravljačkom čvoru podatke o svojoj adresi u fizičkoj mreži te vrsti sadržaja koji je spreman ponuditi ostalim sudionicima mreže

ravnopravnih sudionika. Novoučlanjeni i upravljački čvor zajednički dogovaraju logičko ime koje se pridodijeljuje novom čvoru. Prijavljivanjem pri upravljačkom čvoru novoučlanjeni čvor može definirati sigurnosne uvjete pod kojima ostali sudionici mreže mogu pristupati sadržaju kojim raspolaže. Sigurnosni uvjeti uključuju potrebu za provjerom identiteta pristupnih čvorova, provjeru njihovih prava pristupa, postupke i ključeve korištene za uspostavljanje sigurne komunikacije i slično. Nakon što je postupak ulaska u mrežu završio, upravljački čvor prepozna novoučlanjeni čvor kao jedan od ravnopravnih sudionika mreže. Ostali članovi mreže ravnopravnih sudionika u tom trenutku još uvijek ne znaju za postojanje novog čvora u mreži. Čvor saznaće za postojanje novog čvora u mreži tek nakon što iz mreže zatraži sadržaj koji je dostupan na novoučlanjenom čvoru. Postupak pretraživanja sadržaja u mješovitoj mreži ravnopravnih sudionika prikazan je slikom 3.15.



Slika 3.15 Postupak pretraživanja sadržaja u mješovitoj mreži ravnopravnih sudionika

U postupku traženja sadržaja dostupnog u mreži, čvor šalje upit upravljačkom čvoru (1). Upravljački čvor pretraživanjem svojih podataka o čvorovima pronalazi čvor koji raspolaže zahtijevanim sadržajem. Informacije o imenu i adresi odredišnog čvora upravljački čvor šalje čvoru koji je postavio zahtjev (2). Nakon toga pristupni čvor može poslati zahtjev za sadržajem odredišnom čvoru (3), koji odgovara na zahtjev slanjem traženog sadržaja (4). Ako za odredišni čvor postoje definirani sigurnosni uvjeti za uspostavljanje komunikacije, njihovu provjeru može potaknuti upravljački ili odredišni čvor. Ako je provjera sigurnosnih uvjeta potaknuta od strane upravljačkog čvora, tada upravljački čvor u odgovor pristupnom čvoru uključuje i jamstvo s njegovim pravima pristupa. Ako je provjera sigurnosnih uvjeta potaknuta od strane odredišnog čvora, tada se odredišni čvor po primitku zahtjeva za sadržajem obraća upravljačkom čvoru s ciljem provjere sigurnosnih uvjeta pristupnog čvora. U oba slučaja provjera sigurnosnih uvjeta obavlja upravljački čvor. Pretraživanje sadržaja i provjera sigurnosnih uvjeta u mješovitim mrežama ravnopravnih sudionika obavlja se, dakle, posredstvom upravljačkog čvora, dok se razmjena sadržaja obavlja uspostavljanjem izravne

komunikacijske veze između čvora koji zahtijeva sadržaj i čvora koji tim sadržajem raspolaže.

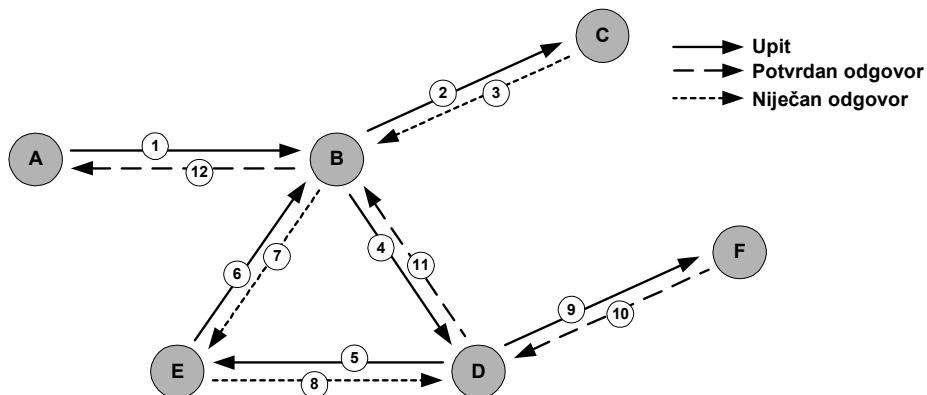
Dinamičnost mješovite mreže ravnopravnih sudionika održava upravljački čvor. On je jedini element u sustavu koji u svakom trenutku zna za postojanje svih sudionika mreže, njihove adrese u fizičkoj mreži, sadržaj kojim raspolaže te sigurnosne uvjete pod kojima se ostvaruje pristup do tog sadržaja. Ostali sudionici mreže doznaju za postojanje drugih čvorova u mreži tek po slanju upita upravljačkom čvoru. Zbog središnjeg mjesta upravljanja oblikom i ustrojem mreže, članstvom sudionika u mreži, pretraživanjem i usmjeravanjem sadržaja te upravljanjem sigurnošću komunikacije, programski međusloj mreže ravnopravnih sudionika moguće je ostvariti jednostavnim algoritmima. Nedostaci arhitekture mješovite mreže ravnopravnih sudionika dolaze do izražaja u mrežama s velikim brojem sudionika, kada dolazi do pojave zagušenja središnjeg upravljačkog čvora. Osnovna prednost arhitekture mješovite mreže ravnopravnih sudionika je jednostavnost izvedbe programskog međusloja, dok ograničena potpora razmernom rastu sustava predstavlja njezin glavni nedostatak.

Upravljanje mrežom zasnovano na preplavljivanju mreže

Znatno složenije postupke upravljanja oblikom i ustrojem mreže, pretraživanja i usmjeravanja sadržaja te provjere sigurnosnih uvjeta od onih primjenjivih u mješovitim mrežama potrebno je provoditi u čistim mrežama ravnopravnih sudionika. Složenost ovih postupaka u tim je mrežama povećana zbog nepostojanja središnjeg upravljačkog elementa. Dok je za mješovite mreže ravnopravnih sudionika svojstveno raspodijeljeno uređenje sadržaja sa središnjim mjestom upravljanja mrežom, čiste mreže ravnopravnih sudionika obilježava potpuna raspodijeljenost, kako sadržaja, tako i upravljačkih postupaka.

Upravljanje mrežom zasnovano na preplavljivanju mreže upitima za pretraživanje sadržaja je raspodijeljeni postupak koji ravnopravni sudionici mreže provode samostalno, bez potrebe za postojanjem središnjeg upravljačkog elementa. Postupak preplavljivanja mreže upitima za pretraživanje sadržaja primjenjuje se u čistim nestrukturiranim mrežama ravnopravnih sudionika. Tijekom ulaska u mrežu novi se čvor povezuje s nekoliko već učlanjenih čvorova. Tijekom napuštanja mreže potrebno je voditi računa o mogućem razdvajanju mreže na više nepovezanih dijelova. Ako se čvor koji napušta mrežu nalazi na jednom putu koji povezuje dva dijela mreže, prije njegova odlaska iz mreže potrebno je dodatno međusobno povezati njegove susjedne čvorove.

Pretraživanje i usmjeravanje sadržaja zasnovano na preplavljivanju mreže koristi načelo uzastopnog slanja upita za sadržajem svim susjednim čvorovima. Čvor redom odašilje istovjetne upite svim susjednim čvorovima sve dok neki od njih ne pošalje poruku da raspolaže zahtijevanim sadržajem ili dok svi čvorovi ne pošalju poruku da traženi sadržaj nije pronađen. Postupak pretraživanja sadržaja postupkom preplavljivanja mreže uzastopnim upitima prikazan je slikom 3.16. Čvor A šalje upit čvoru B jer je on njegov jedini susjedni čvor (1). Čvor B ne raspolaže sadržajem koji traži čvor A i zato nastavlja sa slanjem upita svim svojim susjednim čvorovima. Za svaki upit koji prosljeđuje susjednim čvorovima, čvor stvara njegovu privremenu presliku kako bi se sprječilo beskonačno kruženje upita u slučaju postojanja petlji u mreži. Nakon stvaranja preslike upita, čvor B ga prosljeđuje čvoru C (2). Kako čvor C ne raspolaže sadržajem koji se upitom zahtjeva niti ga ima kome dalje prosljediti, on čvoru B šalje niječni odgovor na postavljeni upit (3). Čvor B nastavlja sa slanjem upita svom sljedećem susjedu, čvoru D (4). Čvor D u nedostatku zahtijevanog sadržaja prosljeđuje upit čvoru E (5). Slijedeći isti postupak, čvor E ga prosljeđuje čvoru B (6). U tom trenutku čvor B otkriva da je upit obišao zatvorenu petlju u mreži uspoređujući ga s vlastitim lokalnim preslikama. Kako bi sprječio beskonačno kruženje upita mrežom, čvor B šalje čvoru E niječni odgovor (7). Čvor E nema dalnjih susjednih čvorova i vraća niječni odgovor čvoru D (8). Čvor D nastavlja sa slanjem upita čvoru F, koji je njegov jedini preostali susjedni čvor (9). Čvor F raspolaže sadržajem koji se traži te ga šalje čvoru D u povrdnom odgovoru (10). Čvorovi D (11) i B (12) prosljeđuju traženi sadržaj do čvora A.



Slika 3.16 Postupak pretraživanja sadržaja u čistoj mreži ravnopravnih sudionika zasnovan na preplavljivanju mreže

Iako znatno složeniji od pretraživanja sadržaja uporabom središnjeg kazala smještenog na upravljačkom čvoru, prednost opisanog postupka pretraživanja i usmjeravanja sadržaja zasnovanog na postupku preplavljivanja mreže upitima još je uvijek relativno jednostavna programska izvedba. Na učinkovitost postupka pretraživanja i usmjeravanja sadržaja

metodom preplavljanja značajno utječe gustoća povezanosti mreže. Ako je svaki čvor mreže ravnopravnih sudionika povezan s velikim brojem susjednih čvorova, pretraživanje metodom preplavljanja uzrokuje velike količine nesvrishodno generiranog mrežnog prometa te značajna kašnjenja u pribavljanju traženog sadržaja. Moguća poboljašanja metode preplavljanja uključuju uporabu priručnih spremnika (engl. *cache memory*) u koje se sprema sadržaj na njegovom povratnom putu od čvora koji njima raspolaže do čvora koji ga je zatražio. Spremanjem sadržaja u privremene spremnike moguće je značajno skratiti vrijeme i smanjiti broj potrebnih upita tijekom sljedećih zahtjeva za istim sadržajem.

Upravljanje mrežom zasnovano na funkciji sažimanja sadržaja

Nedostaci upravljanja mrežom ravnopravnih sudionika zasnovanog na središnjem upravljačkom čvoru s kazalom sadržaja i čvorova na kojima je sadržaj raspoloživ te na postupku preplavljanja mreže upitima, uspješno su riješeni postupcima upravljanja zasnovanim na funkciji sažimanja sadržaja (engl. *hash function*). Upravljanje mrežom uz uporabu funkcije sažimanja sadržaja znatno je složenije od dvaju prethodno opisanih postupaka, ali pruža znatno bolja svojstva glede učinkovitosti, potpore dinamičnosti mreže te potpore razmernom rastu broja mrežnih čvorova i količine sadržaja. Postupak upravljanja mrežom zasnovan na funkciji sažimanja sadržaja primjenjuje se u čistim strukturiranim mrežama ravnopravnih sudionika.

Osnovna računska operacija na kojoj se zasniva rad programskog međusloja ovakvih vrsta mreža ravnopravnih sudionika je matematička funkcija sažimanja sadržaja. Svakom čvoru mreže ravnopravnih sudionika i svakom sadržaju raspoloživom u mreži dodijeljuje se ključ. Ključ čvora je rezultat primjene funkcije sažimanja nad podacima koji su svojstveni čvoru, dok se ključ sadržaja dobiva kao rezultat primjene funkcije sažimanja nad podacima koji su svojstveni sadržaju. Primjerice, ključ čvora moguće je dobiti primjenom funkcije sažimanja nad adresom čvora, dok je ključ sadržaja moguće dobiti primjenom funkcije sažimanja nad imenom datoteke sadržaja, ključnih riječi sadržaja i slično. Osnovna zamisao u postupku upravljanja oblikom i ustrojem mreže te pretraživanju i usmjeravanju sadržaja sastoji se u pridruživanju sadržaja onom čvoru čiji je ključ jednak ključu sadržaja. Ako takav čvor ne postoji u mreži, sadržaj se pridružuje čvoru s najbližim ključem.

Svaki čvor mreže održava skup komunikacijskih veza sa skupom ostalih čvorova mreže. Algoritmi za odabir komunikacijskih veza prema ostalim čvorovima mreže nastoje udovoljiti zahtjevu da broj susjednih čvorova i broj koraka potrebnih za usmjeravanje poruke do odredišnog čvora imaju relativno male vrijednosti. U postupku usmjeravanja sadržaja

koristi se raspodijeljena tablica usmjeravanja. Svaki čvor mreže održava dio tablice usmjeravanja. U tablici usmjeravanja svakog čvora nalaze se zapisi o ključevima susjednih čvorova i njihovim mrežnim adresama. Čvor koji zaprimi zahtjev usporeduje ključ sadržaja s vlastitim ključem i ključevima susjednih čvorova. Ako je ključ lokalnog čvora najbliži ključu sadržaja, onda je lokalni čvor krajnje odredište sadržaja. Ako je zaprimljen zahtjev za isporukom sadržaja, lokalni čvor započinje s pretraživanjem sadržaja u svom lokalnom spremniku i isporučuje sadržaj čvoru koji je postavio zahtjev. Ako je zaprimljen zahtjev za spremanjem sadržaja, čvor spremi sadržaj u svoj lokalni spremnik. Ako su ključevi nekih od susjednih čvorova bliži ključu sadržaja nego što je ključ lokalnog čvora, lokalni čvor proslijeđuje zahtjev susjednom čvoru s ključem najbližim ključu sadržaja.

Pri promjeni ustroja mreže zbog uključivanja novih čvorova u mrežu i napuštanja mreže od strane postojećih čvorova, potrebno je preuređiti raspodjelu sadržaja po čvorovima i ažurirati tablice usmjeravanja. Algoritmi upravljanja ustrojem mreže ravnopravnih sudionika zasnovani na funkciji sažimanja sadržaja provode preuređivanje raspodjele sadržaja i ažuriranje tablice usmjeravanja na ograničenom broju čvorova. Ako čvor napušta mrežu, sadržaj kojim raspolaze raspodjeljuje se po susjednim čvorovima poštujući pravilo da se pridružuje onim čvorovima čiji su ljučevi najbliži ključu sadržaja. Ako čvor ulazi u mrežu, dio sadržaja sa susjednih čvorova seli se na novoučlanjeni čvor. Svaki susjedni čvor novoučlanjenom čvoru predaje dio sadržaja s ključevima koji su najbliži ključu novoučlanjenog čvora.

3.3 Zgodom oblikovane mreže pokretnih sudionika

Zgodom oblikovane mreže pokretnih sudionika (engl. *mobile ad hoc networks*) [62] jedno su od novijih područja primjene prividnih mreža. Zgodom oblikovane mreže pokretnih sudionika su bežične mreže sastavljene od pokretljivih mrežnih čvorova. Budući da fizičke komunikacijske veze u bežičnim mrežama nisu potrebne, komunikacijsku infrastrukturu potrebnu za rad takvih mrežnih sustava čine pokretljivi mrežni čvorovi i programska potpora koja osigurava njihovu međusobnu komunikaciju. Područje primjene zgodom oblikovanih mreža pokretnih sudionika su umreženi sustavi iznimno visokog stupnja pokretljivosti mrežnih čvorova koji su izloženi vrlo nepredvidljivim radnim uvjetima. Nepredvidljivost radnih uvjeta uključuje neplanirano učlanjivanje čvorova u mrežu i napuštanje mreže, kvarove čvorova, prekide komunikacijskih veza zbog izlaska čvorova iz područja dometa signala drugih čvorova i slično. Osim toga, načela rada zgodom oblikovanih mreža pokretnih sudionika moguće je primjeniti i za izgradnju umreženih sustava koji se uobičajeno

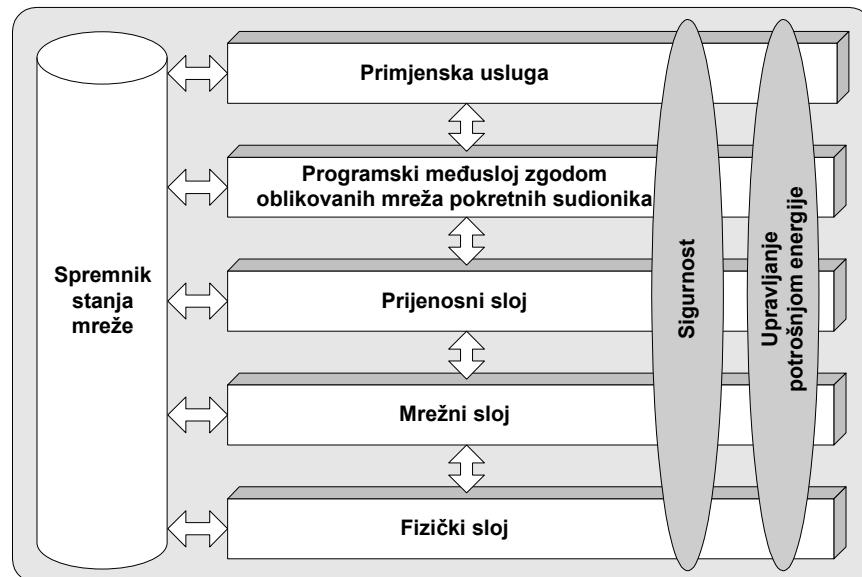
zasnivaju na primjeni ožičenih mreža, ali u okruženjima rada sustava ne postoji izgrađena fizička komunikacijska infrastruktura. Prividnost zgodom oblikovanih mreža pokretnih sudionika očituje se u dinamičkom uspostavljanju bežičnih, odnosno prividnih, komunikacijskih veza među pokretljivim čvorovima mreže za vrijeme rada sustava.

Pri oblikovanju programske potpore koja osigurava rad zgodom oblikovane mreže pokretnih sudionika potrebno je voditi računa o radnim uvjetima u kojima mrežni čvorovi obavljaju zadanu funkciju. Osim bežičnih međusobnih komunikacijskih veza, mrežni čvorovi zgodom oblikovanih mreža pokretnih sudionika uglavnom nisu priključeni na stalni izvor električne energije. Energiju potrebnu za vlastito napajanje mrežni čvorovi crpe iz baterijskih izvora. Jedan od najvažnijih zahtjeva za izgradnju zgodom oblikovanih mreža pokretnih sudionika je učinkovito gospodarenje ograničenim količinama energije. Učinkovitosti gospodarenja potrošnjom električne energije iz baterijskih izvora u najvećoj mjeri pridonosi učinkovita izvedba komunikacije među pokretljivim mrežnim čvorovima. Potrošnja električne energije tijekom prijenosa podataka bežičnim komunikacijskim medijem nekoliko je redova veličine veća od obrade podataka na lokalnom čvoru. Primjerice, za prijenos 100 bitova informacije bežičnom vezom na udaljenosti od 100 metara potrebno je utrošiti približno $10 \mu\text{J}$ energije, dok izvođenje 32-bitne instrukcije na procesoru ARM7TDMI zahtijeva samo 0.06 nJ energije [26]. Razlika u potrošnji iznosi više nego 100 000 puta.

Sigurnost komunikacije i nadzor nad pravima pristupa do funkcionalnosti i sadržaja raspoloživih na pojedinim čvorovima mreže je, uz gospodarenje potrošnjom energije, druga neophodna funkcionalnost zgodom oblikovanih mreža. U čvrsto povezanim ožičenim mrežama gdje se raspored čvorova i njihove međusobne veze rijetko ili nikada ne mijenjaju, uspostava povjerljivih odnosa (engl. *trust relationships*) među čvorovima uglavnom se zasniva na statički definiranim pravima pristupa. U takvim sustavima je upravljanje provedbom sigurnosnih postupaka uglavnom prepušteno izdvojenom čvoru mreže. Zgodom oblikovane mreže pokretljivih čvorova, međutim, zahtjevaju izgradnju sigurnosnog sustava koji osigurava sigurnost, privatnost i povjerljivost u dinamičnim okolinama bez središnjeg upravljačkog čvora. Dodatni problem pri ostvarivanju sigurne komunikacije u bežičnim mrežama predstavlja nepostojanje čvrstog medija za prijenos podataka. Pri prijenosu podataka zrakom, presretanje komunikacije i ugrožavanje povjerljivosti podataka znatno je lakše nego pri prijenosu podataka žičanim vodovima. Pri bežičnom prijenosu podataka, podaci su vidljivi svim sudionicicima koji se nađu u dometu signala kojim se podaci prenose.

Programska potpora zgodom oblikovanih mreža pokretnih sudionika organizirana je u višeslojni model [5, 63] koji se sastoji od fizičkog, mrežnog i prijenosnog sloja, programskog međusloja te primjenskih usluga. Široka prihvaćenost slojevitog *TCP/IP* modela [9] mrežne programske potpore na kojem se zasniva rad komunikacijskih protokola mreže Internet najvažniji je razlog za primjenu sličnog modela i u zgodom oblikovanim mrežama pokretnih sudionika. Dodatni motiv za odabir slojevite organizacije programske potpore je težnja ka objedinjavanju zgodom oblikovanih mreže pokretnih sudionika objedine s postojećim tehnologijama globalne mreže Internet. Primjena sličnog organizacijskog modela za izgradnju programske potpore olakšava postupak objedinjavanja (engl. *integration*) zgodom oblikovanih mreža pokretnih sudionika i mreže Internet.

Kao što je prikazano slikom 3.17, najniži sloj u slojevitom modelu programske potpore zgodom oblikovanih mreža pokretnih sudionika je fizički sloj. Fizički sloj ostvaruje logiku za ostvarivanje bežičnog prijenosa podataka. U fizičkom sloju ostvarene su funkcionalnosti za pristup sklopolju za odašiljanje i primanje radio signala, kôdiranje informacije te njezino moduliranje na val nositelj (engl. *carrier*).



Slika 3.17 Slojevita organizacija programske potpore zgodom oblikovanih mreža pokretnih sudionika s funkcijama sigurnosti i upravljanja potrošnjom energije provedenima u svim slojevima

Mrežni sloj ostvaruje funkcionalnosti otkrivanja komunikacijskih putova između izvorišnog i odredišnog čvora i usmjeravanje sadržaja tim putovima. Komunikacija među pokretnim čvorovima zgodom oblikovane mreže odvija se uspostavljanjem neizravnih komunikacijskih veza (engl. *multihop communication*). Izvorišni čvor šalje poruku odredišnom čvoru putem većeg broja čvorova posrednika. Usmjeravanjem poruke u više

skokova (engl. *hop*) putem nekoliko čvorova posrednika uspostavlja se komunikacijski put sa znatno manjom potrošnjom energije nego što bi bilo potrebno utrošiti za izravno odašiljanje signala bežičnom komunikacijskom vezom. Naime, za prijenos radio signala na nekoliko kraćih udaljenosti potrebno je utrošiti znatno manje ukupne energije nego za jedan prijenos na velikoj udaljenosti [26]. Odabir čvorova posrednika na putu od izvorišta do odredišta određen je njihovim položajem u odnosu na položaj izvorišta i odredišta. Otkrivanje komunikacijskih putova i usmjeravanje sadržaja tim putovima u cilju što učinkovitijeg gospodarenja potrošnjom energije osnovni je zadatak mrežnog sloja.

Funkcionalnosti pouzdanog prijenosa podataka između krajnih točaka komunikacije objedinjene su u prijenosnom sloju. Prijenosni sloj jamči isporuku poruka primateljima te sprječava gubitak poruka i višestruke isporuke istih poruka. Gubitak poruka posljedica je dinamičkih promjena komunikacijskih putova zbog pokretljivosti čvorova, dok višestruke isporuke istih poruka nastaju u slučajevima kada je zbog promjene komunikacijskog puta potrebno ponoviti postupak slanja poruke (engl. *retransmission*).

Razvoj primjenskih usluga koje za komunikaciju koriste zgodom oblikovanu mrežu pokretnih čvorova olakšan je funkcionalnostima dostupnima kroz sučelje programskog međusloja. Programska međusloj osigurava kontekstno-svjesnu okolinu za razvoj i izvođenje primjenskih usluga prilagođenu pojedinim područjima primjene. Primjerice, usluge nadzora radnog područja za kretanje pametnih vozila (engl. *intelligent vehicles*), usluge pametnih domova (engl. *smart homes*), usluge upravljanja gradskim prometom i slično zahtijevaju funkcionalnosti samostalne prilagodbe mrežnih čvorova fizičkoj okolini u kojoj se nalaze, komunikacije u stvarnom vremenu i slično.

Dvije najvažnije funkcionalnosti, sigurnost i upravljanje potrošnjom energije, provode se na svim slojevima. Za slojno organiziranu mrežnu programsku potporu uobičajen je prolazak informacije kroz strukturu programskih slojeva u strogo okomitom smjeru u kojem svaki sloj preuzima i predaje informacije samo svojim susjednim slojevima. Nakon što jedan programski sloj završi s obradom informacije u dijelu za koji je zadužen, susjednom sloju se šalje obavijest kako bi on nastavio s dalnjom obradom (engl. *layer triggers*) [63]. Međudjelovanje susjednih slojeva ostvaruje se putem njihovih okomitih sučelja. Radi što učinkovitije provedbe sigurnosnih postupaka i upravljanja potrošnjom energije, zgodom oblikovane mreže pokretnih sudionika koriste mogućnost izravne komunikacije slojeva koji nisu susjedni u slojnoj organizaciji (engl. *cross-layering*) [63]. Primjerice, ako logika primjenske usluge zahtijeva posebno prilagođen postupak usmjeravanja poruka kroz mrežu pokretnih čvorova kako bi se prikrio identitet čvora pošiljatelja i čvora primatelja, najviši

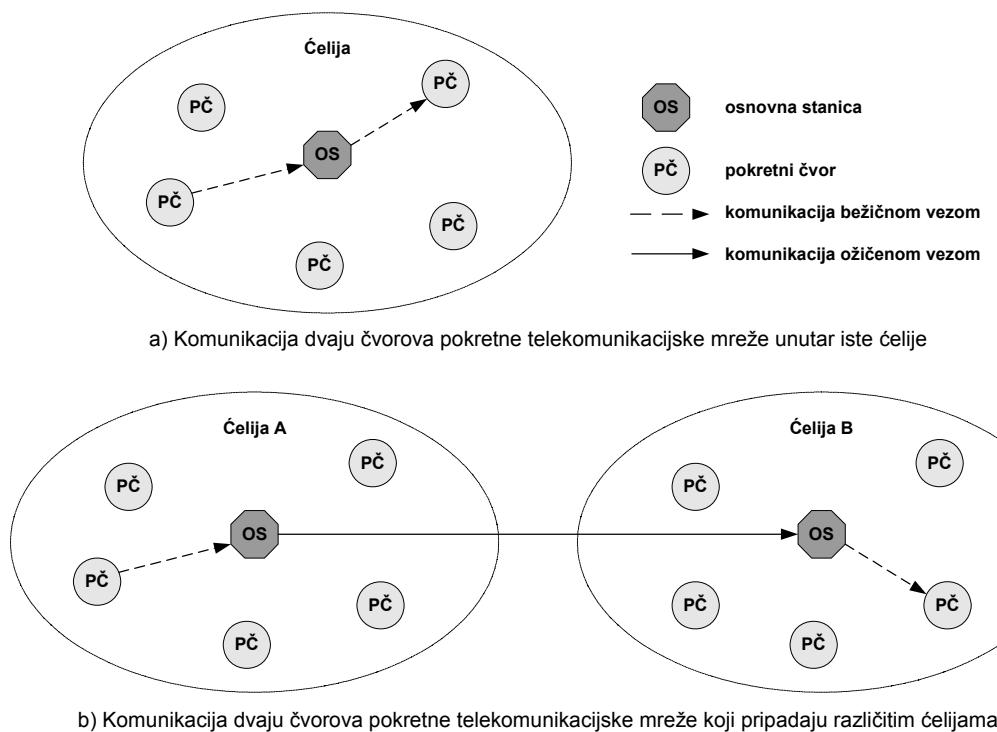
sloj može informacije o usmjerenju predati izravno mrežnom sloju. Za potrebe izravne komunikacije između slojeva koji nisu susjedni, svaki sloj, osim okomitih, izlaže i vodoravna sučelja. Putem vodoravnih sučelja slojevi su povezani sa spremnikom stanja mreže koji sadrži podatke potrebne za učinkovito gospodarenje funkcionalnostima koje se protežu u više slojeva. Upisivanjem podataka u spremnik stanja mreže, sloj uspostavlja izravnu vezu sa slojevima koji mu nisu izravno susjedni. Podaci upisani u spremnik stanja mreže zajednički su svim slojevima. Primjerice, upisivanjem podataka o željenom komunikacijskom putu koji je potrebno slijediti za prikrivanje identita sudionika u komunikaciji, najviši primjenski sloj nalaže mrežnom sloju način usmjerenja podataka.

3.3.1 Upravljanje pokretljivošću sudionika

Zgodom oblikovane mreže pokretnih sudionika po svojoj su prirodi i ponašajnim svojstvima mrežnih čvorova slične pokretnim telekomunikacijskim mrežama (engl. *mobile telecommunication networks*), primjerice GSM (engl. *Global System for Mobile Communications*) mrežama. Sličnost ovih dvaju komunikacijskih sustava očituje se u činjenici što se oba sustava sastoje od pokretljivih mrežnih čvorova koji komunikaciju uspostavljaju odašiljanjem i primanjem radio signala. Oba sustava susreću se i sa sličnim problemima tijekom prijenosa podataka bežičnim komunikacijskim vezama, budući da se krajnji mrežni čvorovi napajaju iz ograničenih baterijskih izvora električne energije.

Razliku između pokretnih telekomunikacijskih mreža i zgodom oblikovanih mreža pokretnih sudionika predstavlja način usmjerenja poruka između čvorova u pokretu. Pokretnе telekomunikacijske mreže zasnivaju se na postojanju osnovne stanice (engl. *base station*) koja služi kao posrednik u komunikaciji između pokretljivih čvorova. Osnovna stanica pokretnih telekomunikacijskih mreža često se naziva i usmjernikom za potporu pokretljivosti čvorova (engl. *mobility support router*). Prostorno područje koje je obuhvaćeno dometom radio signala jedne osnovne stanice naziva se područjem pokrivenosti ili ćelijom (engl. *cell*). Svi pokretljivi mrežni čvorovi koji se nalaze unutar područja pokrivenosti jedne osnovne stanice nazivaju se članovima ćelije. Komunikacija između osnovne stanice i pokretljivog čvora mreže odvija se uspostavljanjem izravne bežične veze (engl. *single hop*). Komunikacija između dvaju pokretljivih čvorova mreže odvija se posredstvom osnovne stanice u dva skoka. U prvom skoku čvor pošiljatelj šalje poruku bežičnom vezom osnovnoj stanici. U drugom koraku osnovna stanica prosljedi poruku bežičnom vezom čvoru primatelju. Ako čvor pošiljatelj i čvor primatelj pripadaju različitim ćelijama, onda osnovna stanica čvora pošiljatelja ne uspostavlja izravnu vezu sa čvorom primateljem, već poruku šalje osnovnoj stanici primateljeve ćelije. Budući da su osnovne

stanice nepokretna postrojenja, priključene su na stalni izvor električne energije koji omogućava prijenos velikih količina podataka bežičnim vezama na velike udaljenosti. Osim toga, komunikacija između osnovnih stanica ne mora nužno biti bežična, već ih je moguće povezati visokopropusnim vodovima za prijenos podataka. Slikom 3.18a prikazan je prijenos podataka između dvaju čvorova pokretne telekomunikacijske mreže unutar iste celije, dok je slikom 3.18b prikazana komunikacija između čvorova koji pripadaju različitim celijama.

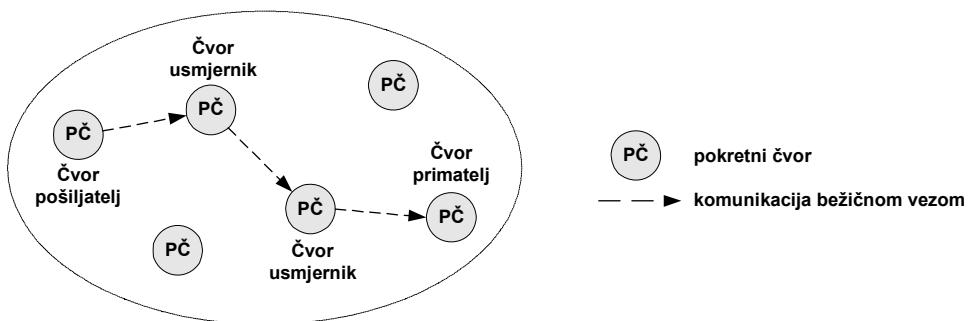


Slika 3.18 Komunikacija između dvaju čvorova pokretne telekomunikacijske mreže bežičnom vezom

Za razliku od pokretnih telekomunikacijskih mreža, u zgodom oblikovanim mrežama pokretnih sudionika ne postoji osnovna stanica koja bi pružala potporu pokretljivosti mrežnih čvorova i omogućavala njihovu međusobnu bežičnu komunikaciju. Pokretljivi mrežni čvorovi zgodom oblikovane mreže međusobne komunikacijske veze uspostavljaju samostalno. Tijekom upućivanja poruke udaljenom odredišnom čvoru koji je izvan dometa radio signala izvorišnog čvora, poruka se usmjerava putem odgovarajućeg broja drugih istovrsnih čvorova (engl. *multihop*). Model prijenosa poruke bežičnom komunikacijskom vezom između pokretljivih čvorova zgodom oblikovane mreže prikazan je slikom 3.19.

U pokretnim telekomunikacijskim mrežama pokretljivi mrežni čvorovi mogu poprimati dvije različite uloge. U trenutku kada čvor šalje poruku drugom čvoru, on poprima ulogu pošiljatelja. U vrijeme primanja poruke od drugog čvora, čvor zauzima ulogu

primatelja. Čvorovi zgodom oblikovane mreže poprimaju tri različite uloge. Osim uloge pošiljatelja i primatelja, čvorovi zgodom oblikovane mreže poprimaju i ulogu usmjernika poruka u slučajevima kada se putem njih prenose radio signali namjenjeni drugim čvorovima mreže. Stoga je usmjeravanje poruka u dinamičnim okruženjima koje čine pokretljivi čvorovi jedan od najvažnijih zadataka programske potpore zgodom oblikovanih mreža pokretnih sudionika.



Slika 3.19 Komunikacija između dvaju čvorova zgodom oblikovane mreže pokretnih sudionika bežičnom vezom

3.3.2 Usmjeravanje sadržaja

Budući da je u smislu potrošnje energije bežična komunikacija najzahtjevniji element zgodom oblikovanih mreža pokretnih sudionika, usmjeravanju poruka u mreži pokretnih čvorova pridaje se značajna pažnja. Usmjeravanje poruka u zgodom oblikovanim mrežama pokretnih sudionika obuhvaća postupke koje je potrebno provesti u slučajevima kada se čvor nađe izvan dometa radio signala drugog čvora. U klasičnim ožičenim mrežama promjene u prostornom rasporedu čvorova vrlo su rijetke, a veze među čvorovima čvrsto definirane uspostavljenim ožičenim vezama. Stoga, usmjeravanje poruka u takvim mrežama uglavnom obuhvaća postupke odabiranja najpovoljnijih komunikacijskih putova od izvorišnog do odredišnog čvora u ovisnosti o zagušenju komunikacijskih veza na pojedinim dijelovima mreže. U zgodom oblikovanim mrežama pokretnih sudionika usmjeravanje ima dvojaku ulogu. Traženjem najpovoljnijeg puta za skokovit prijenos radio signala od izvorišnog do odredišnog čvora upravlja se potrošnjom oskudnih izvora energije. Međutim, osim podizanja razine učinkovitosti, usmjeravanje je potrebno provoditi i zbog održavanja minimalnih radnih uvjeta za uspostavljanje komunikacijskih veza. Minimalni radni uvjeti podrazumijevaju postojanje barem jednog pokretnog čvora u dometu radio signala promatranog pokretnog čvora. Ako promatrani pokretni čvor izade izvan dometa radio signala svih čvorova putem kojih je ostvarivao vezu s ostatkom mreže, slanje podataka potrebno je preusmjeriti na skup čvorova u čijem se dometu trenutno nalazi. Tek ako se

promatrani čvor nalazi u dometu radio signala većeg broja čvorova, moguće je pristupiti odabiru onog čvora koji nudi najpovoljnije uvjete slanja signala bežičnom vezom.

Postupci usmjeravanja poruka u zgodom oblikovanim mrežama pokretnih sudionika svrstavaju se u dvije osnovne skupine. Prvu skupinu čine postupci usmjeravanja s održavanjem komunikacijskih putova u stalnoj pripravnosti (engl. *proactive routing*), dok drugu skupinu čine postupci usmjeravanja s traženjem komunikacijskog puta na zahtjev (engl. *reactive routing*). Svaka od dviju osnovnih skupina postupaka usmjeravanja ima određenih prednosti i nedostataka. Najučinkovitiji postupak usmjeravanja postiže se mješovitim postupcima koji iskorištavaju najbolja svojstva jedne i druge osnovne skupine. Jedan od mješovitih postupaka je postupak usmjeravanja s ranim otkrivanjem prekida veze (engl. *preemptive routing*).

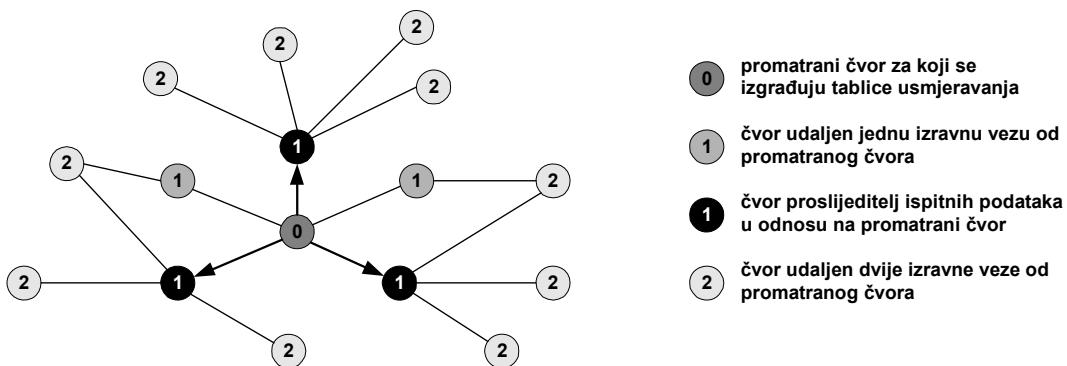
Usmjeravanje s održavanjem komunikacijskih putova u stalnoj pripravnosti

Postupci usmjeravanja poruka u zgodom oblikovanim mrežama pokretnih sudionika s održavanjem komunikacijskih putova u stalnoj pripravnosti nastoje u svakom trenutku svakom čvoru mreže osigurati najsvježiju informaciju o rasporedu čvorova u mreži te skupu čvorova putem kojih je moguće odaslati radio signal drugim čvorovima. U trenutku slanja poruke, izvođeni čvor ima raspoloživu informaciju o tome kojem čvoru je potrebno prenijeti signal bežičnom vezom kako bi se podaci prenijeli do odredišnog čvora. Svaki čvor mreže održava tablicu usmjeravanja koja sadrži zapise o usmjeravanju poruka do ostalih čvorova mreže. Zbog održavanja vjerodostojnih informacija u tablici usmjeravanja, čvorovi mreže povremeno razmjenjuju poruke za produljivanje valjanosti postojećih zapisa u tablici ili za njihovu izmjenu.

Postupak koji se primjenjuje za održavanje vjerodostojnosti tablice usmjeravanja naziva se usmjeravanjem s ispitivanjem stanja veze (engl. *link state routing*) [9] i sličan je algoritmu koji se primjenjuje za razmjenu informacija za usmjeravanje sadržaja u globalnoj mreži Internet. U zgodom oblikovanim mrežama pokretnih sudionika koristi se prilagođena inačica postupka nazvana optimiranim usmjeravanjem s ispitivanjem stanja veze (engl. *optimized link state routing*) [64]. U postupku usmjeravanja s ispitivanjem stanja veze koji se primjenjuje u ožičenim mrežama zasnovanim na IP mrežnom protokolu, mrežni čvorovi povremeno razmjenjuju informacije o zagušenosti komunikacijskih veza. Ispitivanje zagušenosti komunikacijskih veza na pojedinim dijelovima mreže provodi se preplavljivanjem mreže (engl. *broadcasting*) ispitnim podacima i mjeranjem vremena odziva

na pojedinim vezama. Ovisno o vremenu odziva, pojedine se komunikacijske veze vrednuju kao više ili manje pogodne za prijenos korisnih podataka.

Postupak usmjeravanja s ispitivanjem stanja veze uzrokuje razmjenjivanje značajnih količina mrežnog prometa potrebnog za održavanje minimalnih radnih uvjeta mrežnog sustava. U optimiranoj inačici tog postupka, preplavljanje mreže porukama s ispitnim podacima i ispitivanje stanja svih veza zamjenjuje se slanjem ispitnih podataka samo po određenom skupu komunikacijskih veza. Ispitni podaci u zgodom oblikovanim mrežama pokretnih sudionika sadrže informacije o prostornom rasporedu čvorova i njihovim međusobnim komunikacijskim vezama. Svaki čvor mreže odabire vlastiti skup čvorova proslijeditelja (engl. *multipoint relay*). Odabir čvorova proslijeditelja obavlja se iz skupa susjednih čvorova. Uloga čvora proslijeditelja je proslijđivanje ispitnih podataka sebi susjednog čvora prema ostalim čvorovima mreže te proslijđivanje korisnih podataka drugih čvorova prema svom susjednom čvoru. Iz skupa svih susjednih čvorova, čvor odabire minimalni broj čvorova proslijeditelja putem kojih može ostvariti vezu sa svim čvorovima koji su od njega udaljeni dvije izravne komunikacijske veze. Slikom 3.20 prikazan je prostorni raspored mrežnih čvorova dijela zgodom oblikovane mreže s istaknutim čvorovima proslijediteljima u odnosu na središnji promatrani čvor.



Slika 3.20 Odabir čvorova proslijeditelja prilikom razmjene informacija za usmjeravanje sadržaja postupkom s održavanjem komunikacijskih putova u stalnoj pripravnosti

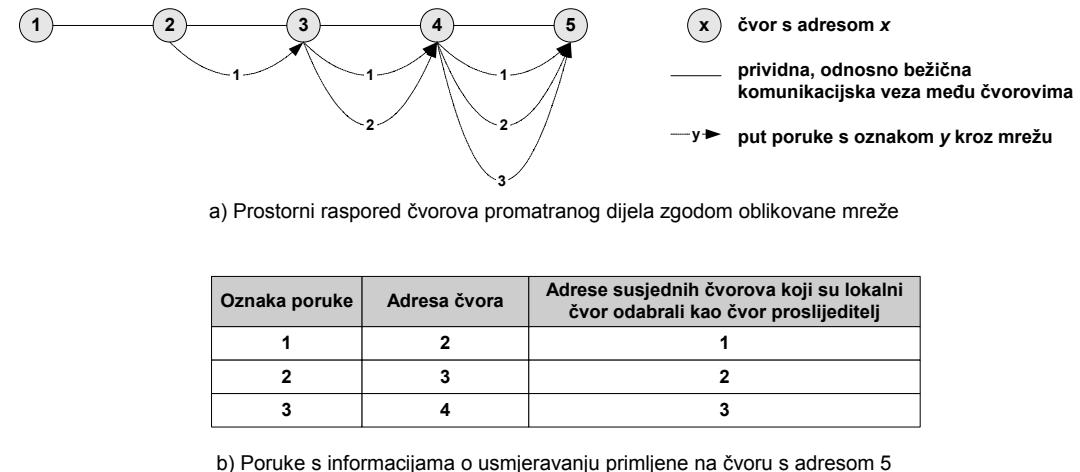
Informacije o prostornom rasporedu čvorova prikupljaju se povremenim odašiljanjem pozdravnih poruka. Budući da je komunikacija bežična, pozdravnu poruku primaju svi čvorovi koji su u dometu signala čvora pošiljatelja. U pozdravnoj poruci čvor pošiljatelj navodi listu poznatih susjednih čvorova te listu susjednih čvorova koje je odabrao kao čvorove proslijeditelje. Čvor mreže šalje pozdravnu poruku nakon isteka određenog vremenskog razmaka, odnosno nakon što razmjenom pozdravnih poruka s drugim čvorovima mreže utvrdi promjene u prostornom rasporedu čvorova. Promjene u prostornom rasporedu

utječu na skup susjednih čvorova te na skup susjednih čvorova koji su odabrani kao čvorovi proslijeditelji. Na osnovi razmjene pozdravnih poruka svaki čvor doznaće skup sebi susjednih čvorova, skup čvorova koji su od njega udaljeni dvije izravne komunikacijske veze te skup susjednih čvorova putem kojih može ostvariti vezu sa svakim od čvorova koji je od njega udaljen dvije izravne komunikacijske veze. Analizom podataka o susjednim čvorovima koje prima u pozdravnim porukama, čvor određuje vlastiti skup čvorova proslijeditelja, što također dojavljuje svojim susjednim čvorovima u pozdravnim porukama. Iz podataka sadržanih u pozdravnim porukama koje prima od drugih čvorova, čvor gradi listu susjednih čvorova koji su upravo njega odabrali kao čvor proslijeditelj. Svaki čvor na osnovi razmjenjenih pozdravnih poruka gradi tri tablice: tablicu svih susjednih čvorova, tablicu susjednih čvorova koje je odabrao za čvorove proslijeditelje i tablicu susjednih čvorova koji su njega odabrali kao čvor proslijeditelj.

Na osnovi tablice susjednih čvorova koji su njega odabrali kao čvor proslijeditelj gradi se tablica usmjeravanja sadržaja. Ako čvor mreže utvrdi promjene u skupu čvorova od kojih je odabran kao čvor proslijeditelj, čvor odašilje poruku za razmjenu informacija o usmjeravanju sadržaja. Porukom za razmjenu informacija o usmjeravanju sadržaja čvor pošiljatelj svim čvorovima u dometu vlastitog signala dojavljuje podatke o skupu čvorova od kojih je odabran kao čvor proslijeditelj. Primljena poruka s informacijama o usmjeravanju koja sadrži podatke o skupu čvorova od kojih je čvor pošiljatelj odabran kao čvor proslijeditelj koristi se za izgradnju tablice usmjeravanja. Svaki čvor koji primi poruku provjerava, i po potrebi obnavlja, postojeće zapise u vlastitoj tablici usmjeravanja. Podaci sadržani u primljenoj poruci govore da je čvor koji je poslao poruku posljednji čvor u lancu čvorova putem kojih se usmjeravaju korisni podaci (engl. *last hop*) do čvorova koji su njega odabrali za čvor usmjeritelj. Unutrašnjim pretraživanjem po podacima iz svih primljenih poruka, čvor primatelj dolazi do adrese prvog čvora u lancu putem kojeg je potrebno usmjeravati promet do pojedinog odredišnog čvora. Primjer na slici 3.21 prikazuje postupak izgradnje pravila usmjeravanja sadržaja od čvora 5 do čvora 1 na osnovi tri primljene poruke s informacijama o usmjeravanju sadržaja.

Po primitku poruke za razmjenu informacija o usmjeravanju sadržaja, svaki čvor koji ju je primio provjerava da li se nalazi u skupu čvorova proslijeditelja onog čvora od kojeg je poruka pristigla. Ako se nalazi u tom skupu, nastavlja s dalnjim odašiljanjem poruke. Ako se čvor koji je primio poruku ne nalazi u skupu čvorova proslijeditelja, prekida daljnje slanje poruke. Budući da samo oni čvorovi koji se nalaze u skupu čvorova proslijeditelja čvora primatelja nastavljaju sa širenjem informacija o usmjeravanju, u mreži se stvara znatno

manja količina mrežnog prometa u odnosu na neoptimirani postupak usmjeravanja s ispitivanjem stanja veze.



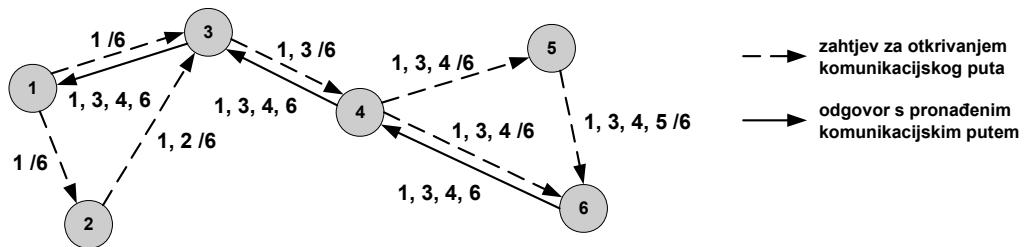
Slika 3.21 Postupak razmjene informacija o usmjeravanju sadržaja te izradnja pravila usmjeravanja na osnovi prikupljenih podataka

Čvor 2 u prvoj poruci šalje informaciju da je on odabran kao čvor proslijeditelj od strane čvora 1. Poruku prima čvor 3 te je prosljeđuje dalje, budući da je on čvor proslijeditelj za čvor 2. Slijedeći isti postupak, čvor 4 prosljeđuje poruku do čvora 5. Sličnim načinom razmjene poruka s informacijama o usmjeravanju sadržaja čvor 5 saznaće da je čvor 3 čvor proslijeditelj za čvor 2 te da je čvor 4 čvor proslijeditelj za čvor 3. Za izgradnju pravila usmjeravanja do čvora 1, čvor 5 iskorištava sve tri primljene poruke u postupku unutražnjog pretraživanja komunikacijskog puta. Na osnovi poruke s oznakom 1, čvor 5 zaključuje da je sadržaj moguće uputiti čvoru 1 usmjeravanjem putem čvora 2. Budući da čvor 2 nije susjedni čvor čvoru 5, postupak unutražnjog pretraživanja se nastavlja. Potrebno je pronaći put do čvora 2 jer putem njega vodi put do čvora 1. Iz poruke s oznakom 2 dobiva se podatak da je slanje sadržaja do čvora 2 moguće obaviti usmjeravanjem putem čvora 3. Budući da ni čvor 3 nije susjedni čvor čvoru 5, postupak se nastavlja traženjem puta do čvora 3. Poruka s oznakom 3 sadrži podatak da je slanje sadržaja do čvora 3 moguće obaviti usmjeravanjem putem čvora 4. Čvor 4 je susjedni čvoru 5 te čvor 5 u vlastitu tablicu usmjeravanja upisuje pravilo kojim se sadržaj namijenjen čvoru 1 usmjerava putem čvora 4. Osim pravila usmjeravanja do čvora 1, ovim je postupkom čvor 5 zaključio kako je za usmjeravanje sadržaja namijenjenog čvorovima 2 i 3 također potrebno koristiti čvor 4.

Usmjeravanje s traženjem komunikacijskog puta na zahtjev

Usmjeravanje sadržaja s održavanjem komunikacijskih putova u stalnoj pripravnosti uzrokuje značajne količine mrežnog prometa potrebnog za održavanje povezanosti mreže. U slučajevima vrlo dinamičke okoline u kojoj mrežni čvorovi učestalo mijenjaju položaj, razmjena informacija o usmjeravanju u znatnoj mjeri opterećuje komunikacijske putove te troši ograničene baterijske izvore energije. Za primjenu u takvim okolinama osmišljeni su postupci usmjeravanja sadržaja s traženjem komunikacijskog puta na zahtjev. Postupci usmjeravanja sadržaja u zgodom oblikovanim mrežama pokretnih sudionika s traženjem komunikacijskog puta na zahtjev nemaju unaprijed pripremljenu informaciju o rasporedu čvorova u mreži niti o mogućim komunikacijskim putovima kroz mrežu. Odašiljanju radio signala s izvorišnog čvora s informacijom za odredišni čvor prethodi postupak otkrivanja komunikacijskog puta kroz mrežu čvorova od izvorišnog do odredišnog čvora.

Usmjeravanje s traženjem komunikacijskog puta na zahtjev moguće je provesti na više načina, koristeći nekoliko razvijenih algoritama. Jedan od često primjenjivanih postupaka je dinamičko usmjeravanje pod nadzorom izvorišta podataka (engl. *dynamic source routing*). U postupku dinamičkog usmjeravanja pod nadzorom izvorišta podataka, izvorišni čvor određuje cijelokupni komunikacijski put kojim sadržaj prolazi na putu od izvorišta, putem više čvorova umjernika, do odredišta. Budući da se prostorni raspored čvorova učestalo mijenja, slanju sadržaja prethodi određivanje komunikacijskog puta. Slijed čvorova kojima prolazi sadržaj na putu od izvorišta do odredišta određuje se na zahtjev, bez potrebe za povremenim izmjenama informacija o usmjeravanju među čvorovima mreže kojima se komunikacijski putovi održavaju pripravnima.



Slika 3.22 Postupak usmjeravanja sadržaja u zgodom oblikovanim mrežama pokretnih sudionika s traženjem komunikacijskog puta na zahtjev

Komunikacijski put određuje se preplavljivanjem mreže porukama sa zahtjevima za otkrivanjem komunikacijskog puta. Postupak prikupljanja informacije o raspoloživom putu od izvorišnog do odredišnog čvora prikazan je slikom 3.22. Izvorišni čvor odašilje signal s porukom za traženjem komunikacijskog puta do odredišnog čvora. U poruku ugrađuje

vlastitu adresu i adresu odredišnog čvora. Svi čvorovi koji prime poruku sa zahtjevom za otkrivanjem komunikacijskog puta nadopunjaju poruku vlastitom adresom te je proslijedju dalje svim svojim susjednim čvorovima. Čvor koji utvrđi da je upravo on odredišni čvor, odgovara porukom s pronađenim komunikacijskim putom. Poruka s odgovorom sadrži adresu svih čvorova kojima je prolazila poruka sa zahtjevom za otkrivanjem komunikacijskog puta. Na taj način izvorišni čvor dobiva potpunu informaciju o slijedu čvorova kojim treba uputiti željeni sadržaj. U prikazanom primjeru na slici 3.22, čvor 1 predstavlja izvorišni čvor, dok je odredište smješteno na čvoru 6. Čvor 1 odašilje signal s porukom za traženjem komunikacijskog puta do čvora 6. U poruku ugrađuje vlastitu adresu 1. Čvorovi 2 i 3 koji su u dometu signala čvora 1 primaju poruku, proširuju je vlastitim adresama i odašilju nove poruke sa zahtjevima. Osim adrese čvorova kojima prolazi, poruka sa zahtjevom sadrži i redni broj poruke. Redni broj poruke služi za otkrivanje višestruko primljenih poruka s istim zahtjevom. Primjerice, čvor 3 najprije od čvora 1 prima poruku za zahtjevom za otkrivanje komunikacijskog puta od čvora 1 do čvora 6. Tu poruku proslijeduje dalje jer on nije krajnje odredište poruke. Nakon toga čvor 3 prima od čvora 2 poruku s istim zahtjevom koja je izvorno potekla od čvora 1. Čvor 3 tada prekida daljnje slanje poruke. Slijedeći opisani postupak, zahtjev konačno stiže do čvora 6, koji utvrđuje da je on njezino kranje odredište i priprema poruku s odgovorom koji sadrži informacije o pronađenom komunikacijskom putu od čvora 1 do čvora 6. Čvor 6 poruku s odgovorom vraća onim putem koji mu je ona i pristigla, a isti postupak provode i svi ostali čvorovi na unatražnom putu do izvorišnog čvora 1. Po primitku odgovora, čvor 1 ima potpunu informaciju o slijedu čvorova kojima je potrebno proslijedivati sadržaj do odredišnog čvora. Svi čvorovi koji imaju ulogu usmjernika na putu od čvora 1 do čvora 6, odnosno čvorovi 3 i 4, također imaju potrebne informacije o slanju sadržaja do čvora 6 te ne moraju započinjati novi postupak otkrivanja komunikacijskog puta do tog čvora.

Usmjeravanje sadržaja s ranim otkrivanjem prekida veze

Osnovni nedostatak postupka usmjeravanja sadržaja u zgodom oblikovanim mrežama pokretnih sudionika s održavanjem komunikacijskih putova u stalnoj pripravnosti jest učestala komunikacija među mrežnim čvorovima radi razmjene informacija o njihovom međusobnom prostornom rasporedu. Učestala komunikacija potrebna za održavanje radnih uvjeta takvog mrežnog sustava ima nepovoljan utjecaj na gospodarenje potrošnjom energije. Učinkovitije gospodarenje potrošnjom energije postiže se primjenom postupka usmjeravanja sadržaja s otkrivanjem komunikacijskog puta na zahtjev. Njegov je nedostatak, međutim, potreba za pronalaskom komunikacijskog puta prije svakog slanja sadržaja drugom čvoru.

Dodatni problemi javljaju se tijekom dugotrajnih veza. Ako veza između dvaju čvorova treba biti uspostavljena duže vrijeme, otkrivanje komunikacijskog puta prije njezina uspostavljanja ne jamči da taj put neće biti prekinut za vrijeme komunikacije uslijed kretanja čvorova. Rješenje problema ovih dvaju postupaka usmjeravanja nudi postupak usmjeravanja sadržaja s ranim otkrivanjem prekida veze (engl. *preemptive routing*) [66].

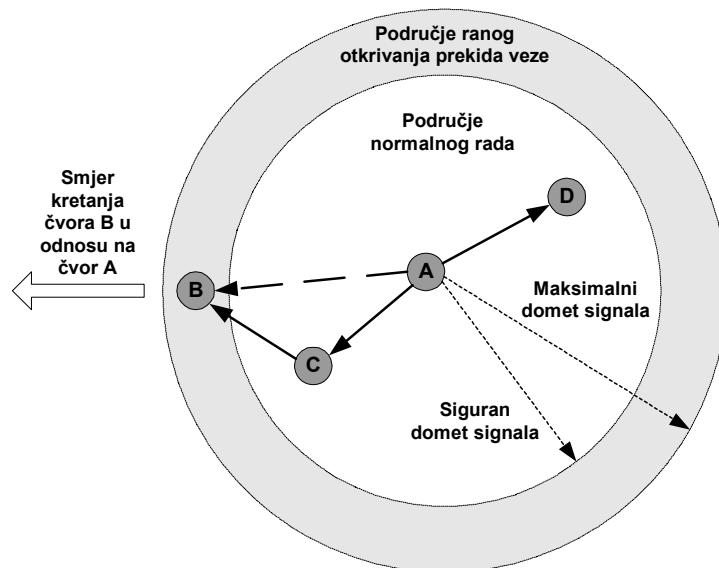
Postupak usmjeravanja sadržaja s ranim otkrivanjem prekida veze mješoviti je postupak koji iskorištava dobra svojstva postupka usmjeravanja s održavanjem komunikacijskih putova u stalnoj pripravnosti i postupka usmjeravanja s otkrivanjem komunikacijskog puta na zahtjev, a potiskuje njihove najveće nedostatke. Okosnicu ovog postupka usmjeravanja čini usmjeravanje s pronalaženjem komunikacijskog puta na zahtjev, ali su uvedena određena proširenja. Usmjeravanje sadržaja obavlja se u dvije osnovne faze. Prva faza sastoji se od otkrivanja komunikacijskog puta do čvora s kojim se nastoji uspostaviti komunikacija. Drugu fazu postupka čini održavanje pronađenog komunikacijskog puta u pripravnosti.

Prva faza postupka usmjeravanja provodi se prije slanja korisnih podataka od izvorišnog prema odredišnom čvoru. Za provedbu prve faze zadužen je izvorišni čvor. Prije slanja korisnih podataka s izvorišnog na odredišni čvor, izvorišni čvor započinje postupak otkrivanja komunikacijskog puta do odredišnog čvora. Otkrivanje komunikacijskog puta do odredišnog čvora obavlja se na zahtjev te se provodi prethodno opisanim postupkom za usmjeravanje sadržaja s otkrivanjem komunikacijskog puta na zahtjev. Nakon što je komunikacijski put do odredišnog čvora pronađen, izvorišni čvor podatke o pronađenom putu koristi za slanje korisnih podataka.

Pronalaskom komunikacijskog puta do odredišnog čvora započinje druga faza postupka usmjeravanja sadržaja. Budući da su čvorovi pokretljivi, za vrijeme komunikacije, odnosno za vrijeme slanja korisnih podataka od izvorišnog prema odredišnom čvoru, postoji izgledna vjerojatnost promjene prostornog rasporeda čvorova. Pri promjeni njihovog međusobnog prostornog rasporeda, čvorovi putem kojih vodi komunikacijski put od izvorišnog do odredišnog čvora mogu izaći izvan dometa radio signala čvorova s kojima ostvaruju izravnu vezu. U tom se slučaju komunikacijska veza između izvorišnog i odredišnog čvora prekida. Prekid veze uzrokuje gubitak podataka koji su u tom trenutku bili na putu od izvorišta prema odredištu. Gubitak podataka je okriven od strane mehanizama prijenosnog sloja programske potpore zgodom oblikovanih mreža, ali je za njihovo ponovno slanje potrebno pronaći novi komunikacijski put kroz mrežu. U cilju preduhithivanja prekida

komunikacijskih veza te uklanjanja potrebe za otkrivanjem novih komunikacijskih putova osmišljen je postupak usmjeravanja sadržaja s ranim otkrivanjem prekida veze.

Usmjeravanje sadržaja s ranim otkrivanjem prekida veze prikuplja podatke o kretanju čvorova u prostoru te njihovom međusobnom prostornom rasporedu i udaljenostima. Na osnovi prikupljenih podataka nastoji se otkriti mogućnost pojave prekida veze prije nego što se on stvarno dogodi te na vrijeme pronaći zamjenski komunikacijski put. Nakon što je pronađen zamjenski komunikacijski put, izvorišni čvor nastavlja slati podatke odredišnom čvoru tim pravcem.



Slika 3.23 Načelo rada postupka usmjeravanja sadržaja s ranim otkrivanjem prekida veze

Načelo rada postupka usmjeravanja s ranim otkrivanjem prekida veze prikazano je slikom 3.23. Područje dometa radio signala izvorišnog čvora dijeli se na područje normalnog rada i područje ranog otkrivanja prekida veze. Cjelokupno područje dometa signala određeno je maksimalnim dometom signala izvorišnog čvora. Područje normalnog rada određeno je udaljenošću koja predstavlja siguran domet signala. Ova je udaljenost uvijek manja od maksimalnog dometa signala, a utvrđuje se na osnovi stupnja slabljenja signala zbog prepreka u radnoj okolini mreže i slično. Izvorišni čvor može nesmetano komunicirati izravnim bežičnim vezama sa svim čvorovima koji se nalaze unutar njegovog područja normalnog rada. Ako neki od čvorova s kojima izvorišni čvor komunicira izravnom vezom uđe u područje otkrivanja prekida veze, komunikacija je još uvijek moguća, ali postoje naznake o međusobnom udaljavanju izvorišnog i odredišnog čvora, što bi moglo dovesti do prekida veze i gubitka podataka. U cilju sprječavanja nasilnog prekida veze i gubitka podataka, odredišni čvor mjeri jakost signala izvorišnog čvora. Kada jakost signala padne

ispod unaprijed određene granice, odredišni čvor zaključuje da je ušao u područje ranog otkrivanja prekida veze. Odredišni čvor tada šalje poruku upozorenja izvorišnom čvoru. Po primitku poruke upozorenja, izvorišni čvor započinje postupak otkrivanja zamjenskog komunikacijskog puta. Na slici 3.23 čvor A na početku ima uspostavljenu izravnu vezu s čvorom B. Čvor B se, međutim, udaljava od čvora A i signal čvora A koji prima čvor B sve je slabiji. Ulaskom u područje ranog otkrivanja prekida veze, čvor B šalje poruku upozorenja čvoru A. Pokretanjem postupka otkrivanja novog komunikacijskog puta, čvor A pronalazi zamjenski put do čvora B putem čvora C, koji se još uvijek nalazi u području normalnog rada čvora A. Čvor C je odabran kao posrednik u komunikaciji između čvorova A i B iz razloga što se čvor B nalazi u području normalnog rada čvora C.

Postupak usmjeravanja s ranim otkrivanjem prekida veze nastoji pronaći zamjenski komunikacijski put u trenutku dok je slanje podataka prvobitno uspostavljenim putem još uvijek moguće. Prebacivanjem komunikacije na zamjenski komunikacijski put u trenutku dok je prvobitno uspostavljeni put još uvijek u funkciji sprječava se gubitak podataka koji nastaje uslijed prekida veze na prvobitno uspostavljenom putu. Istodobno se uklanja i vrijeme čekanja potrebno za pronalazak novog komunikacijskog puta kroz mrežu.

4 Programski međusloj zasnovan na razmjeni poruka

Razvojem raspodijeljenog računarstva, primjena raspodijeljenih programskih sustava širila se na sve veći broj područja ljudske djelatnosti. Rast raspodijeljenih programskih sustava po brojnosti i po zemljopisnoj rasprostranjenosti iziskivao je i uvodenje određenih promjena u radu komunikacijskih podsustava koji omogućuju komunikaciju među sastavnim dijelovima raspodijeljenog sustava. Radno okruženje suvremenih raspodijeljenih sustava su raznorodne okoline, razvijene u različitim programskim jezicima, zasnovane na različitim programskim i sklopovskim tehnologijama te podložne različitostima u određivanju organizacijskih pravila i načina upravljanja. Većina područja primjene raspodijeljenih sustava zahtjeva neprekidni rad te visoku razinu kakvoće usluge (engl. *Quality of Service – QoS*) i sigurnosti (engl. *security*). Tradicionalni način izgradnje raspodijeljenih programskih sustava zasnovan na komunikaciji pozivima udaljenih procedura ne može odgovoriti zahtjevima visokog stupnja dostupnosti, kakvoće pružanja usluge i razmjernog rasta raspodijeljenog sustava.

Odgovor na izazove razvoja komunikacijskih podsustava složenih raspodijeljenih računalnih sustava ponuđen je osmišljavanjem programskog međusloja zasnovanog na razmjeni poruka (engl. *Message Oriented Middleware – MOM*) [6, 7, 8]. Programski međusloj zasnovan na razmjeni poruka je programski sustav koji omogućava komunikaciju među sastavnim dijelovima raspodijeljenog sustava slanjem poruka. Iskorištavanjem funkcionalnosti programskog međusloja zasnovanog na razmjeni poruka, dijelovi raspodijeljenog sustava mogu slati i primati poruke od drugih dijelova raspodijeljenog sustava. U nastavku poglavlja opisana su osnovna načela rada programskog međusloja zasnovanog na razmjeni poruka te su istaknute ključne razlike u svojstvima raspodijeljenih sustava zasnovanih na programskom međusloju za razmjenu poruka i onih zasnovanih na mehanizmima poziva udaljenih procedura.

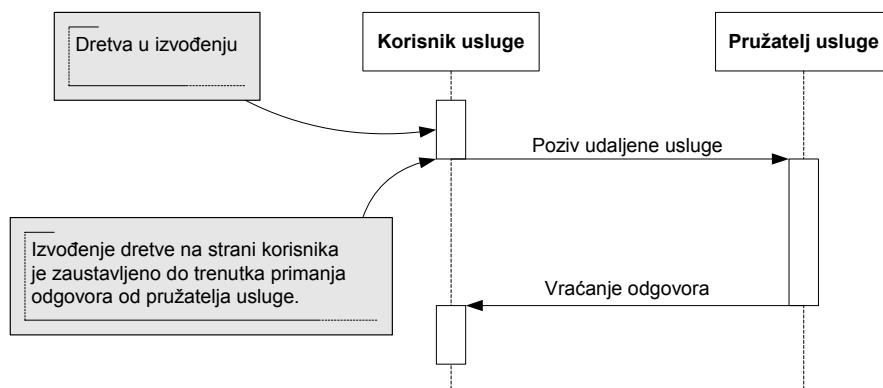
4.1 Modeli međudjelovanja u komunikacijskim sustavima

Za izgradnju suvremenih raspodijeljenih računalnih sustava koriste se dva osnovna modela komunikacijskog međudjelovanja: sinkrona i asinkrona komunikacija. Sinkrona komunikacija pogodna je za primjenu u raspodijeljenim računalnim sustavima u kojima je moguće ostvariti vremensku istodobnost postojanja i raspoloživosti pošiljatelja i primatelja informacije. Komunikacijski model zasnovan na asinkronoj komunikaciji omogućuje vremensku razdvojenost pošiljatelja i primatelja informacije. Dok se u slučaju sinkrone

komunikacije uspostavlja izravna veza između pošiljatelja i primatelja, u slučaju asinkrone komunikacije veza se uspostavlja putem komunikacijskog posrednika. Komunikacijski posrednik je element komunikacijskog sustava koji omogućuje vremensku razdvojenost pošiljatelja i primatelja informacije.

4.1.1 Sinkrona komunikacija

Elementi komunikacijskog sustava koji koriste sinkroni model komunikacijskog međudjelovanja uspostavljaju izravnu komunikacijsku vezu. Uspostava izravne komunikacijske veze zahtijeva istodobno postojanje i raspoloživost svih elemenata sustava uključenih u komunikacijski postupak. Sinkroni komunikacijski model svojstven je pozivima potprograma u proceduralnim, odnosno pozivima metoda nad objektima u objektno orijentiranim programskim jezicima. U raspodijeljenim sustavima, model sinkrone komunikacije koriste sustavi zasnovani na korisnik-poslužitelj arhitekturi te sustavi zasnovani na mehanizmima poziva udaljenih procedura. Najrašireniji raspodijeljeni sustav zasnovan na sinkronom komunikacijskom modelu jest *World Wide Web* [9]. Budući da pošiljatelj informacije obično poziva primatelja s ciljem obavljanja određene usluge, članovi komunikacijskog sustava nazivaju se pozivateljem ili korisnikom usluge, odnosno pružateljem usluge.



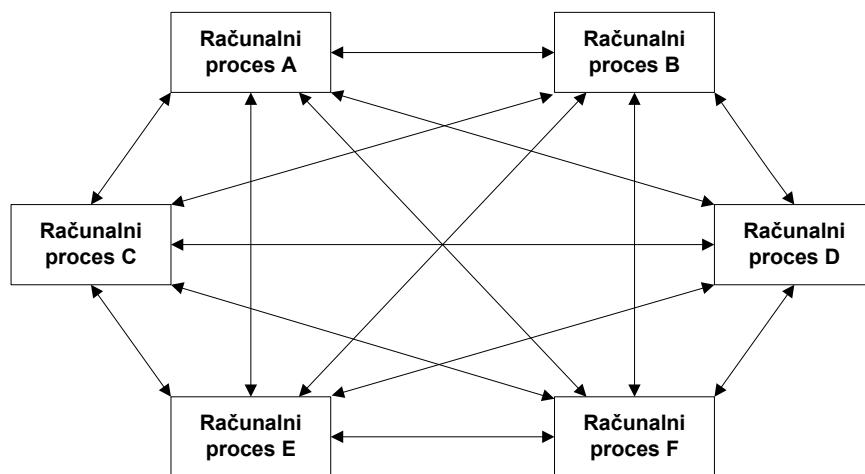
Slika 4.1 Model međudjelovanja zasnovan na sinkronoj komunikaciji

Osnovno svojstvo sinkronog komunikacijskog modela je vremensko isključivanje izvođenja korisnika i pružatelja usluge. U trenutku poziva usluge, korisnik zaustavlja vlastitu dretvu (engl. *thread*) i izvođenje se prenosi na stranu pružatelja usluge. Nakon što pružatelj obradi korisnikov zahtjev i vrati korisniku rezultate poziva usluge, korisnik nastavlja s izvođenjem vlastite dretve. Model međudjelovanja zasnovan na sinkronoj komunikaciji prikazan je slikom 4.1. Iako su računalni procesi povezani sinkronim komunikacijskim

modelom prostorno raspodijeljeni, sinkrona komunikacija ne dozvoljava njihovo istodobno izvođenje.

Pozivi udaljenih procedura

Najčešće korišteno programsko ostvarenje sinkronog komunikacijskog modela je mehanizam poziva udaljenih procedura (engl. *Remote Procedure Call – RPC*) [6, 7, 8]. Mehanizam poziva udaljenih procedura razvila je tvrtka *Sun Microsystems Inc.* tijekom osamdesetih godina dvadesetog stoljeća. Većina današnjih posredničkih sustava za potporu razvoju i izvođenju raspodijeljenih programskega sustava, kao što su *CORBA*, *Java RMI*, *Microsoft DCOM* i *XMLRPC* [8], koristi upravo taj mehanizam. Uloga posredničkog sustava zasnovanog na mehanizmu poziva udaljenih procedura je omogućavanje pozivanja programskega procedura koje se izvode u računalnim procesima na udaljenim mrežnim čvorovima.



Slika 4.2 Sustav raspodijeljenih računalnih procesa koji međusobno komuniciraju modelom poziva udaljenih procedura

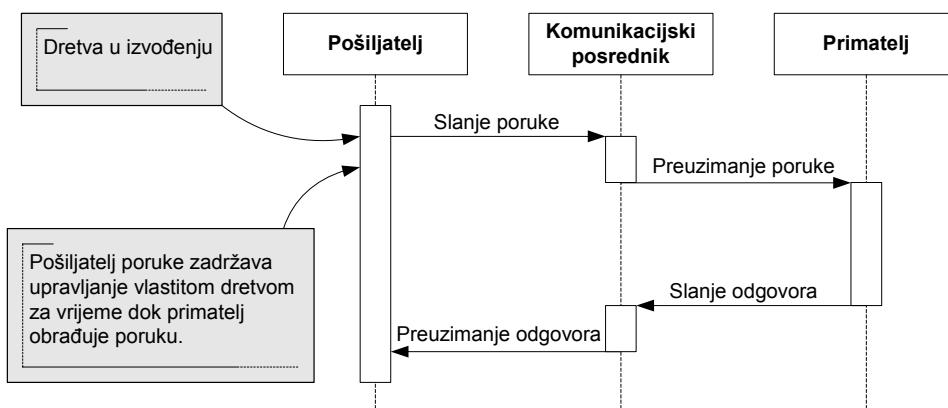
Tijekom komunikacije raspodijeljenih računalnih procesa koji koriste mehanizam poziva udaljenih procedura, jedan od procesa poziva proceduru koja se izvodi u drugom, udaljenom procesu. S gledišta istodobnosti izvođenja dvaju procesa koji komuniciraju, poziv udaljene procedure sličan je pozivu procedure unutar istog procesa. Pozivajući proces zaustavlja svoje izvođenje i predaje upravljanje nad izvođenjem procedure udaljenom procesu. Izvođenje pozivajućeg procesa ostaje zaustavljeno sve dok se ne primi povratna vrijednost poziva procedure od udaljenog procesa. Koristeći se mehanizmom poziva udaljenih procedura, raspodijeljeni računalni procesi prividno komuniciraju kao da su dio

istog procesa. Komunikacijska infrastruktura posredničkog sustava prikriva od komunicirajućih procesa njihovu raspodijeljenost te moguće različitosti u tehnologijama njihovog programskog ostvarenja, okruženjima u kojima se izvode i načinu prikaza podataka koji koriste.

Slikom 4.2 prikazan je primjer raspodijeljenog računalnog sustava čiji sastavni dijelovi međusobno komuniciraju primjenom mehanizma poziva udaljenih procedura. Između svakog para procesa koji komuniciraju postoji izravna komunikacijska veza. U trenutku poziva udaljene procedure, pozvani proces mora biti aktivan i spreman izvršiti pozvanu proceduru.

4.1.2 Asinkrona komunikacija

Za razliku od sinkronog modela međudjelovanja među sastavnim dijelovima raspodijeljenog računalnog sustava, model zasnovan na asinkronoj komunikaciji omogućuje raspodijeljeno i istodobno izvođenje procesa koji sudjeluju u komunikaciji. U slučaju asinkrone komunikacije, pošiljatelj i primatelj informacije ne uspostavljaju izravnu komunikacijsku vezu, već se ta veza uspostavlja posredstvom komunikacijskog posrednika (engl. *broker*). Komunikacijski posrednik je dio programskog međusloja zasnovanog na razmjeni poruka koji osigurava sigurni i pouzdani prijenos te pravodobnu isporuku poruka između procesa u komunikaciji. Komunikacijski posrednik omogućuje vremensku razdvojenost pošiljatelja i primatelja informacije. Vremenskim razdvajanjem pošiljatelja i primatelja omogućeno je istodobno izvođenje raspodijeljenih procesa te mogućnost komunikacije bez potrebe istovremenog postojanja i raspoloživosti obaju procesa u komunikaciji. Slikom 4.3 prikazan je model međudjelovanja zasnovan na asinkronoj komunikaciji.

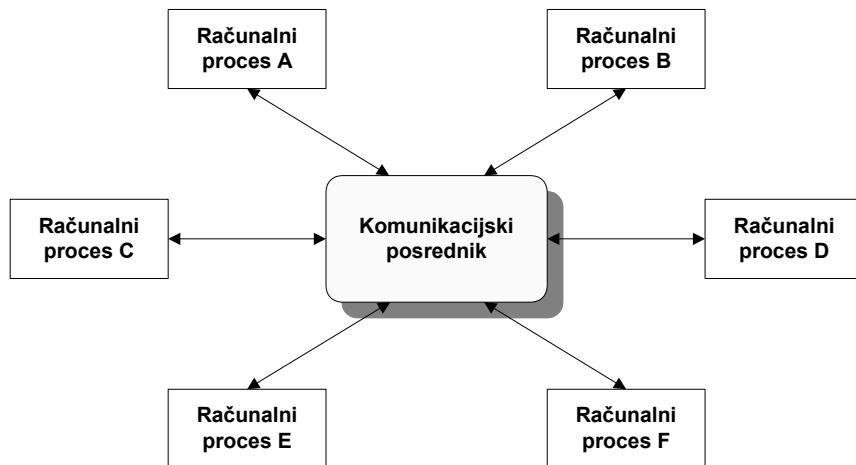


Slika 4.3 Model međudjelovanja zasnovan na asinkronoj komunikaciji

Umjesto izravnog upućivanja poruke primatelju, pošiljatelj šalje poruku komunikacijskom posredniku. Komunikacijski posrednik brine o isporuci poruke primatelju. Ako je na primljenu poruku potrebno odgovoriti, primatelj preuzima ulogu pošiljatelja i upućuje poruku s odgovorom komunikacijskom posredniku. Pošiljatelj izvorne poruke preuzima odgovor od komunikacijskog posrednika. Uloga komunikacijskog posrednika je pouzdana isporuka poruke primatelju te pravodobno obavještavanje izvornog pošiljatelja o pristiglom odgovoru. Budući da pošiljatelj poruke ne treba voditi brigu o preuzimanju odgovora od primatelja, on može nastaviti s obavljanjem onih dijelova vlastite dretve za koje nije potreban odgovor primatelja. Time je, osim raspodijeljenosti, omogućena i istodobnost izvođenja procesa pošiljatelja i procesa primatelja. S druge strane, ako primatelj nije u mogućnosti primiti poruku u trenutku slanja, komunikacijski posrednik može je zadržati za kasniju isporuku. Pošiljatelj poruke zaustavlja napredovanje vlastite dretve tek u trenutku kada je za nastavak izvođenja neophodan odgovor primatelja. Ako odgovor primatelja nije bitan za nastavak izvođenja dretve pošiljatelja, izvođenje procesa pošiljatelja nije onemogućeno privremenom ili stalnom neraspoloživošću primatelja. Na taj je način omogućena vremenska razdvojenost pošiljatelja i primatelja poruke. Najpoznatiji i najrašireniji raspodijeljeni računalni sustav zasnovan na asinkronom komunikacijskom modelu je sustav za razmjenu elektroničke pošte (engl. *electronic mail, e-mail*) [9].

Komunikacija razmjenom poruka

Najčešće korišteno programsko ostvarenje asinkronog komunikacijskog modela je komunikacija razmjenom poruka. Ključna razlika između poziva udaljenih procedura i komunikacije razmjenom poruka je način upravljanja izvođenjem raspodijeljenih dijelova računalnog sustava. Računalni procesi koji komuniciraju razmjenom poruka imaju svojstvo slabe povezanosti (engl. *loose coupling*) [10]. Slaba povezanost procesa omogućuje visok stupanj samostalnosti i međusobne neovisnosti procesa. Svaki se proces izvodi samostalno i neovisno o svim drugim procesima s kojima ostvaruje komunikaciju. Svaki proces tijekom komunikacije s drugim procesom upućuje poruku komunikacijskom posredniku koji brine o sigurnoj, pouzdanoj i pravodobnoj isporuci poruke procesu primatelju. Budući da su procesi slabo povezani, potreban je dodatni mehanizam koji povezuje poruke s procesima kojima su namjenjene. Ako pošiljatelj poruke na poslanu poruku očekuje odgovor od primatelja, dodatnim je mehanizmima potrebno povezati poruke zahtjeva i odgovora koje čine istu logičku cjelinu. Logičko povezivanje fizički razdvojenih procesa obavlja programski međusloj zasnovan na razmjeni poruka koji ima ulogu komunikacijskog posrednika.



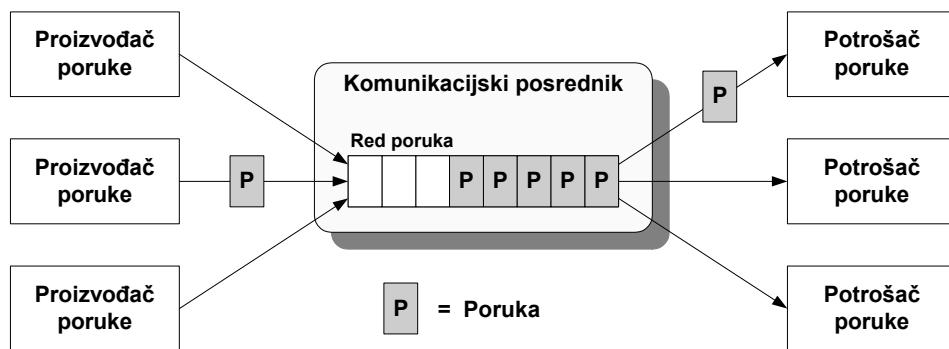
Slika 4.4 Sustav raspodijeljenih računalnih procesa koji međusobno komuniciraju razmjenom poruka

Slikom 4.4 prikazan je sustav raspodijeljenih računalnih procesa čiji sastavni dijelovi međusobno komuniciraju razmjenom poruka. Procesi u komunikaciji ne uspostavljaju izravne komunikacijske veze, već su svi povezani putem komunikacijskog posrednika. U trenutku slanja poruke nije nužno osigurati primateljevu dostupnost i spremnost prihvatanja poruke te slanja odgovora. U slučaju privremene nedostupnosti primatelja, komunikacijski posrednik zadržava poruku za kasniju isporuku. Za vrijeme dok posrednik isporučuje poruku primatelju i dok je primatelj obrađuje, pošiljatelj nastavlja izvođenje vlastitog procesa. Komunikacija zasnovana na razmjeni poruka neophodna je u slučajevima u kojima slanje poruke uzrokuje dugotrajni postupak obrade poruke na strani primatelja, u slučajevima kada primatelj poruke nema uspostavljenu stalnu komunikacijsku vezu s ostalim dijelovima sustava te u sustavima u kojima je za izvršavanje obrade poruke potrebno posredovanje čovjeka. U takvim je okolnostima zaustavljanje izvođenja pošiljatelja i prijenos izvođenja procesa na stranu primatelja neprihvatljivo s gledišta pošiljatelja zbog predugih razdoblja čekanja na završetak obrade na strani primatelja.

Redovi poruka

Osnovno načelo rada većine komunikacijskih posrednika zasnovanih na razmjeni poruka je komunikacija putem redova poruka. Redovi poruka su programske komponente komunikacijskog posrednika koje se koriste za prihvat, spremanje i isporuku zaprimljenih poruka. Koristeći redove poruka, komunikacijski posrednik primjenjuje *spremi-i-isporuči* (engl. *store-and-forward*) mehanizam komunikacije. Komunikacija raspodijeljenih računalnih procesa ostvarena redovima poruka prikazana je slikom 4.5. Računalni procesi koji šalju poruke u red poruka nazivaju se proizvođačima poruka. Procesi koji uzimaju

poruke iz reda poruka nazivaju se potrošačima poruka. Tijekom slanja poruke udaljenom procesu, proces proizvođač šalje poruku komunikacijskom posredniku. Komunikacijski posrednik poruku spremi u red poruka. Zaprimljena poruka ostaje spremljena u redu poruka sve dok neki od procesa potrošača ne zatraži poruku.



Slika 4.5 Komunikacija raspodijeljenih računalnih procesa ostvarena redom poruka

Osim razmjene poruka, komunikacija putem redova poruka osigurava i vremensko usklađivanje izvođenja (engl. *synchronization*) raspodijeljenih računalnih procesa. Ako proces potrošač za napredovanje izvođenja treba pribaviti poruku iz reda poruka, a proces proizvođač poruku još nije poslao, proces potrošač ostaje zaglavljen u redu čekanja. U trenutku pristizanja poruke od procesa proizvođača, komunikacijski posrednik iz reda čekanja osloboda prvog potrošača te mu proslijedi pristiglu poruku. Kako bi se omogućilo vremensko uklajivanje izvođenja procesa, komunikacijski posrednik, osim reda poruka, koristi i red čekanja na poruku.

4.1.3 Analiza primjenjivosti modela međudjelovanja u komunikacijskim sustavima

Oba modela međudjelovanja raspodijeljenih računalnih procesa u komunikacijskim sustavima imaju svojih prednosti i nedostataka, ovisno o načinu i području primjene. Bitna svojstva komunikacijskih modela koja je potrebno razmotriti i usporediti pri donošenju odluke o njihovoj primjeni za izgradnju raspodijeljenog računalnog sustava su jednostavnost programske izvedbe, stupanj zavisnosti raspodijeljenih dijelova računalnog sustava, pouzdanost prijenosa podataka, potpora razmernom rastu sustava, zahtijevani stupanj dostupnosti raspodijeljenih dijelova sustava te potpora istodobnosti izvođenja raspodijeljenih procesa.

Sinkroni komunikacijski model zasnovan na pozivima udaljenih procedura je s gledišta programera jednostavniji za korištenje od asinkronog modela zasnovanog na

razmjeni poruka. Poziv udaljene procedure sličan je uobičajenom i široko prihvaćenom postupku pozivanja lokalnih potprograma u proceduralnom programiranju, odnosno pozivanju metoda nad objektima u objektno orijentiranom programiranju. Programsко ostvarenje raspodijeljenog računalnog sustava primjenom mehanizma poziva udaljenih procedura programeru daje jasniju predodžbu prostorne raspodijeljenosti dijelova sustava i vremenske raspodjele njihova izvođenja. Pozivom udaljene procedure zaustavlja se izvođenje pozivajućeg procesa te se prenosi pozvanom procesu. Po završetku izvođenja udaljene procedure, pozivajući proces dobiva rezultate njezina izvođenja i nastavlja s izvođenjem vlastite dretve. Budući da je komunikacija sinkrona, raspodijeljeni procesi se obavljaju onim redom kojim su pozvane udaljene procedure, a istim redom izvođenje i završava. U slučaju poziva većeg broja udaljenih procedura, one se izvode slijedno. Poziv i izvođenje sljedeće udaljene procedure u zadanim slijedu može započeti tek po završetku izvođenja prethodno pozvane procedure. Asinkroni komunikacijski model zasnovan na razmjeni poruka programeru ostavlja manji stupanj nadzora nad izvođenjem raspodijeljenih dijelova sustava. Upućivanjem poruke primatelju uz posredovanje komunikacijskog posrednika pokreće se izvođenje udaljenog procesa. Upućivanjem poruka većem broju udaljenih procesa moguće je istodobno pokretanje većeg broja udaljenih procesa. Nije, međutim, nužno pokretanje udaljenih procesa onim redoslijedom kojim su im upućivane poruke, kao što nije predvidiv ni redoslijed njihova završavanja.

Dijelovi raspodijeljenog sustava povezani komunikacijskim modelom zasnovanim na pozivima udaljenih procedura su čvrsto povezani računalni procesi. Računalni procesi su izravno povezani točno definiranim sučeljima udaljenih procedura putem kojih se ostvaruje pristup do funkcionalnosti udaljenog procesa. U slučaju potrebe za promjenom sučelja jednog procesa, svi procesi koji s njim ostvaruju komunikacijsku vezu moraju se prilagoditi novom načinu ostvarivanja komunikacije. Prilagodba sučelja uzrokuje privremeno zaustavljanje cjelokupnog sustava. Sustavi koji vezu ostvaruju razmjenom poruka su slabo povezani sustavi jer komunikacijsku vezu ne uspostavljaju izravno, već putem posrednika. Za ostvarivanje komunikacije s udaljenim procesom nije bitan oblik njegovog pristupnog sučelja, nego oblik poruke koji je tom procesu razumljiv. Ako je tumačenje poruka ostvareno pretraživanjem dijelova informacije unutar poruke po ključevima, a ne prema položaju, naknadna proširenja oblika poruke neće imati utjecaja na postojeće procese koji s tim procesom ostvaruju vezu.

Asinkronim komunikacijskim modelom zasnovanim na razmjeni poruka uz primjenu *spremi-i-isporuči* mehanizma jamči se pouzdana isporuka poruke primatelju. Poruke pristigle od pošiljatelja komunikacijski posrednik sprema u lokalnom spremniku. Ako

primatelj poruke u trenutku zaprimanja poruke od pošiljatelja nije dostupan ili nije u mogućnosti primiti poruku, komunikacijski posrednik na određeno vrijeme odgadá prosljeđivanje poruke te povremenim ispitivanjem spremnosti primatelja pokušava obaviti isporuku. Poruka se briše iz posrednikova spremnika tek po uspješnoj isporuci. Pozivi udaljenih procedura ne jamče pouzdanost komunikacije. Ako proces nad kojim se poziva udaljena procedura u trenutku poziva uslijed preopterećenosti, kvara ili sličnih problema nije spreman izvršiti pozvanu proceduru, poziv procedure neće uspjeti. Pozivajući proces se u tom slučaju treba sam pobrinuti za postupak oporavka od pogreške.

Odabir komunikacijskog modela ima značajan utjecaj na ukupno vrijeme izvođenja raspodijeljenog programskog sustava [68] i prikazan je slikom 4.6. U razmatranje se uzima slučaj u kojem je manji dio zadatka potrebno prosljediti na izvođenje udaljenom procesu i prikupiti rezultate njegova izvođenja, a veći dio posla pozivajući proces obavlja lokalno, bez međudjelovanja s udaljenim procesima. Pritom se prepostavlja da za obavljanje lokalnih poslova nisu potrebni povratni rezultati izvođenja zadatka u udaljenom procesu.

Budući da u slučaju primjene sinkronog komunikacijskog modela zasnovanog na pozivima udaljenih procedura nije moguće ostvariti istodobno izvođenje lokalnih i udaljenih procesa, ukupno vrijeme izvođenja zadatka izračunava se primjenom formule (1).

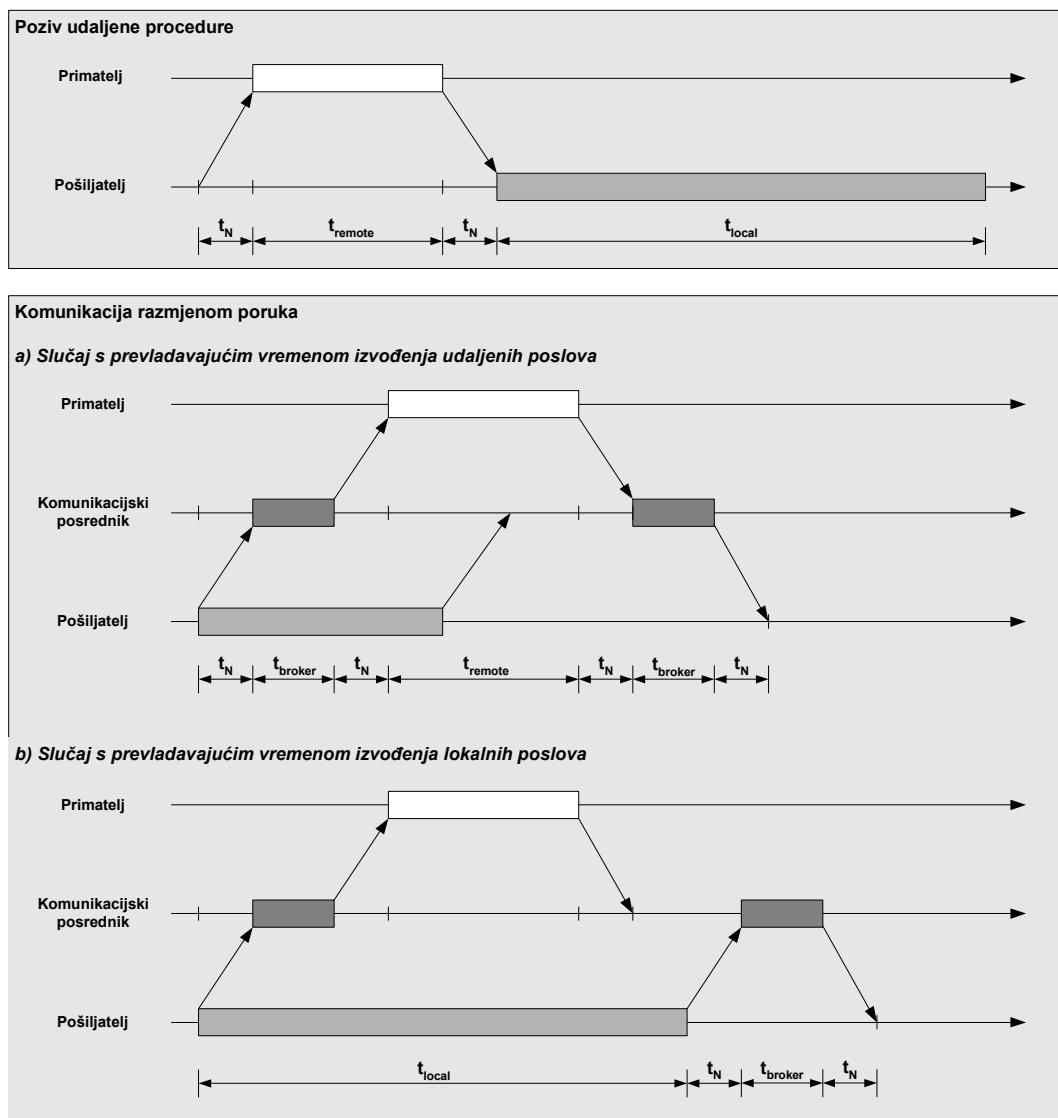
$$t_{RPC} = t_N + t_{remote} + t_N + t_{local} = t_{remote} + t_{local} + 2t_N \quad (1)$$

Ukupno vrijeme izvođenja zadatka pri pozivu udaljene procedure (t_{RPC}) jednako je zbroju trajanja izvođenja udaljene procedure u udaljenom procesu (t_{remote}) i trajanja izvođenja lokalnih poslova (t_{local}), uvećano za vrijeme potrebno za prijenos parametara poziva procedure računalnom mrežom do udaljenog procesa (t_N) i pribavljanje povratnih rezultata (t_N).

Ako se za povezivanje raspodijeljenih procesa upotrijebi asinkroni komunikacijski model zasnovan na razmjeni poruka, onda se izvođenje dijela zadatka u udaljenom procesu može obavljati istodobno s izvođenjem lokalnih poslova za koje nije potreban rezultat izvođenja udaljenog procesa. Ukupno vrijeme izvođenja zadatka ovisi o tome da li je više vremena potrebno za slanje poruke udaljenom procesu te izvođenje dijela zadatka u udaljenom procesu ili je prevladavajuće vrijeme izvođenja onog dijela zadatka koji izvodi lokalni proces. Ako je vrijeme potrebno za slanje poruke i izvođenje zadatka u udaljenom procesu prevladavajuće, ukupno vrijeme izvođenja zadatka određeno je formulom (2).

$$t_{MOM} = t_N + t_{broker} + t_N + t_{remote} + t_N + t_{broker} + t_N = t_{remote} + 2t_{broker} + 4t_N \quad (2)$$

Ukupno vrijeme potrebno za izvođenje zadatka (t_{MOM}) uključuje vrijeme izvođenja zadatka u udaljenom procesu (t_{remote}), vremena potrebna komunikacijskom posredniku za proslijedivanje poruke udaljenom procesu (t_{broker}) te proslijedivanje odgovora pozivajućem procesu (t_{broker}) i mrežnog kašnjenja. Budući da se za ostvarivanje međuprocesne komunikacije koriste dvije poruke i svaka se mrežom prenosi dva puta, ukupno mrežno kašnjenje jednako je četverostrukom iznosu jediničnog mrežnog kašnjenja (t_N).



Slika 4.6. Usporedba utjecaja sinkronog i asinkronog komunikacijskog modela na vrijeme izvođenja raspodijeljenih računalnih procesa

U slučaju prevladavajućeg vremena izvođenja lokalnih poslova, slanje poruke za pokretanje zadatka u udaljenom procesu, izvođenje udaljenog zadatka te vraćanje odgovora komunikacijskom posredniku završava prije nego što završi izvođenje lokalnih poslova.

Nakon što obavi lokalne poslove, pozivajući proces treba pribaviti rezultat izvođenja udaljenog zadatka od komunikacijskog posrednika. Ukupno vrijeme izvođenja je, stoga, određeno formulom (3).

$$t_{MOM} = t_{local} + t_N + t_{broker} + t_N = t_{local} + t_{broker} + 2t_N \quad (3)$$

Na osnovi formula (2) i (3) i slike 4.6 dobiva se izraz za ukupno vrijeme izvođenja raspodijeljenih računalnih procesa povezanih razmjenom poruka. Izraz za ukupno vrijeme izvođenja predstavljen je formulom (4).

$$t_{MOM} = \max\{ (t_{remote} + t_{broker} + 3t_N), (t_{local} + t_N) \} + t_{broker} + t_N \quad (4)$$

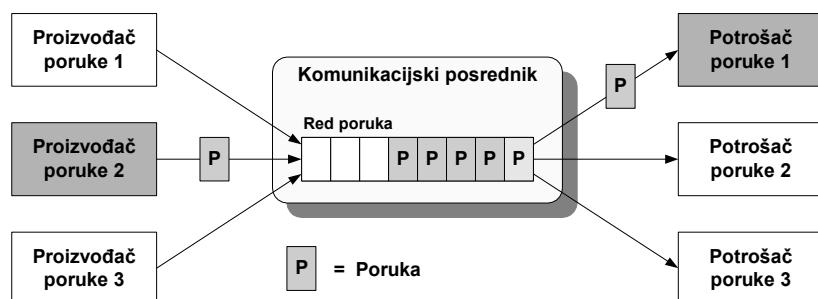
Budući da računalni proces koji poziva udaljenu proceduru drugog procesa mora zaustaviti svoje izvođenje, raspodijeljeni sustavi zasnovani na komunikacijskom modelu poziva udaljenih procedura pružaju ograničene mogućnosti rasta sustava. U slučaju ulančanih poziva udaljenih procedura u kojem pozvana procedura poziva drugu udaljenu proceduru, vrijeme čekanja prvog pozivajućeg procesa u lancu se uvišestručuje. Zbog čvrste povezanosti procesa, brzina rada sustava ograničena je brzinom najsporijeg procesa u sustavu. Usporenje rada zbog opterećenja jednog procesa izravno utječe na smanjenje učinkovitosti cijelokupnog sustava. Povezivanje raspodijeljenih procesa komunikacijskim modelom zasnovanim na razmjeni poruka pruža značajno bolje mogućnosti razmjernog rasta sustava. Budući da svaki proces zadržava upravljanje izvođenjem vlastite dretve, usporenje rada pojedinih procesa nema značajan učinak na svojstva ostatka sustava. S druge strane, procesi preuzimaju poruke od komunikacijskog posrednika tek nakon što su ih spremni obraditi, što ne dovodi do njihove nagle preopterećenosti i zagušenja. Svaki proces samostalno određuje stupanj vlastitog opterećenja, ovisno o vlastitim mogućnostima.

Da bi raspodijeljeni računalni sustav zasnovan na pozivima udaljenih procedura ispravno radio, svi udaljeni procesi moraju u trenutku poziva biti raspoloživi i spremni obaviti pozvanu proceduru. Ako se dogodi ispad, kvar ili je u tijeku nadogradnja nekog od raspodijeljenih procesa, svi ostali procesi koji s njim uspostavljaju komunikaciju neće biti u mogućnosti obavljati svoju normalnu funkciju. U slučaju komunikacije razmjenom poruka, privremeni prestanak rada jednog ili više procesa ne odražava se trenutno na rad ostatka sustava. Utjecaj neispravnosti pozvanog procesa pozivajućem procesu postaje vidljiv tek u trenutku kad su mu bez odgađanja potrebni rezultati izvođenja pozvanog procesa. Međutim, iako sami po sebi potpuno ispravni i funkcionalni, procesi povezani komunikacijskim modelom zasnovanim na razmjeni poruka neće biti u mogućnosti obavljati normalnu funkciju u slučaju kvara na komunikacijskom posredniku.

Na osnovi usporedbe bitnih svojstava dvaju komunikacijskih modela, dolazi se do zaključka da svaki model ima svoja područja primjene. Sinkroni komunikacijski model zasnovan na pozivima udaljenih procedura jednostavniji je za programsko ostvarenje i pruža veći stupanj nadzora nad izvođenjem raspodijeljenih računalnih procesa. S druge strane, dinamički raspodiljeni programski sustavi u kojima nije u svakom trenutku moguće predvidjeti raspoloživost i stupanj opterećenja pojedinih dijelova sustava postižu veću učinkovitost uz primjenu asinkronog modela međudjelovanja. Suvremeni raspodijeljeni programski sustavi sa svojstvom porasta broja sastavnih komponenata i prostorne rasprostranjenosti, zasnovani na raznorodnim tehnologijama te s dinamički promjenjivim odnosima među komponentama zahtijevaju primjenu asinkronog modela međudjelovanja zasnovanog na razmjeni poruka.

4.2 Metode razmjene poruka

Asinkronim komunikacijskim modelom zasnovanim na razmjeni poruka putem redova poruka moguće je ostvariti nekoliko različitih načina komunikacije među raspodijeljenim računalnim procesima. Primjena određene metode komunikacije ovisi o području primjene i zahtjevima koji se postavljaju na komunikacijski sustav. Ovisno o broju potrošača poruke, komunikacijski sustav zasnovan na razmjeni poruka putem redova poruka može biti sustav s jednim ili sustav s više potrošača.

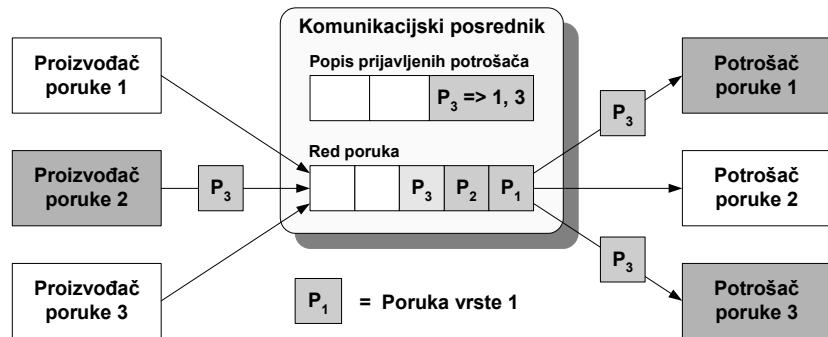


Slika 4.7 Komunikacijski sustav zasnovan na razmjeni poruka putem redova poruka s jednim potrošačem

U sustavu s jednim potrošačem (engl. *point-to-point messaging system*) [8] jedna poruka može biti iskorištena samo od strane jednog potrošača. Nakon što se poruka isporuči prvom potrošaču, ona se briše iz reda poruka. Isporuka poruka iz reda poruka procesima potrošačima najčešće se obavlja po *FIFO* (engl. *First In – First Out*) načelu. Procesu potrošaču isporučuje se poruka koja je prva pristigla u red poruka. Slikom 4.7 prikazan je

komunikacijski sustav zasnovan na razmjeni poruka putem redova poruka s jednim potrošačem. Poruku koju u red poruka šalje proces proizvođač 2 čekaju tri potrošača. Pod pretpostavkom da je proces potrošač 1 prvi zatražio poruku iz reda poruka, komunikacijski posrednik je isporučuje tom procesu. Preostala dva procesa potrošača nastavljaju čekati na pristizanje novih poruka u red poruka.

U sustavu s više potrošača (engl. *publish/subscribe messaging system*) [8] istu je poruku moguće isporučiti većem broju potrošača. Za svaku poruku pristiglu u red poruka vodi se popis procesa koji su prijavljeni za tu vrstu poruke. Pristigla poruka isporučuje se svim prijavljenim procesima. Tek po isporuci poruke svim prijavljenim procesima, poruka se uklanja iz reda poruka. Slikom 4.8 prikazan je komunikacijski sustav zasnovan na razmjeni poruka putem redova poruka s više potrošača. Za poruku vrste 3 koju u red poruka šalje proces proizvođač 2 postoje dva prijavljena potrošača, proces potrošač 1 i proces potrošač 3. Komunikacijski posrednik isporučuje poruku svakom od prijavljenih procesa potrošača.



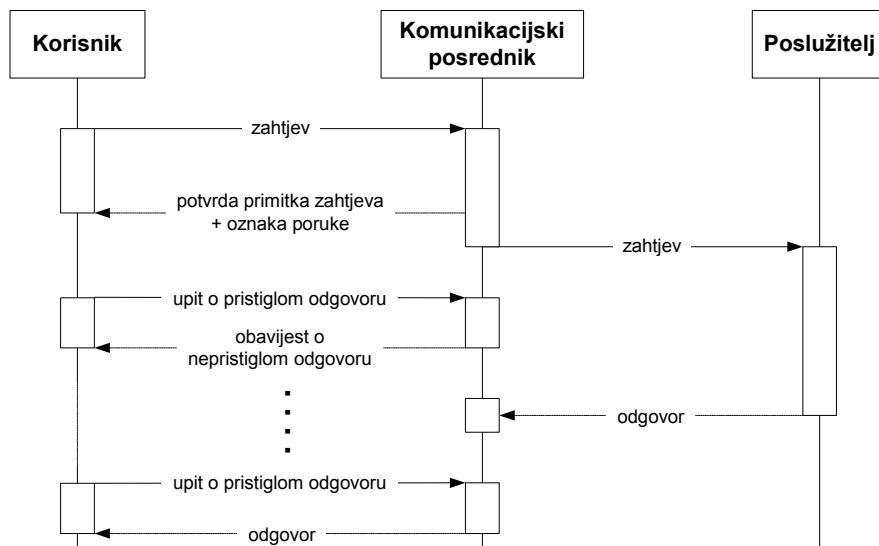
Slika 4.8 Komunikacijski sustav zasnovan na razmjeni poruka putem redova poruka s više potrošača

Osnovne metode za ostvarivanje međuprocesne komunikacije pomoću redova poruka su prozivanje, povratni poziv i objava/preplata. Metode koje koriste prozivanje i povratni poziv primjenjuju se u sustavima s jednim potrošačem. Metoda objava/preplata primjenjiva je u sustavima s više potrošača.

4.2.1 Prozivanje

Najjednostavnija metoda međuprocesne komunikacije zasnovana na razmjeni poruka putem redova poruka je metoda prozivanja (engl. *polling*) [69]. Prozivanje se koristi za komunikaciju procesa među kojima vrijede odnosi korisnika i poslužitelja. U komunikaciji dvaju raspodijeljenih procesa, jedan od procesa ima ulogu poslužitelja koji na korištenje izlaže određenu uslugu. Drugi proces ima ulogu korisnika koji pozivanjem procesa

poslužitelja koristi izloženu uslugu. Komunikacija računalnih procesa uz uporabu metode prozivanja prikazana je slikom 4.9.

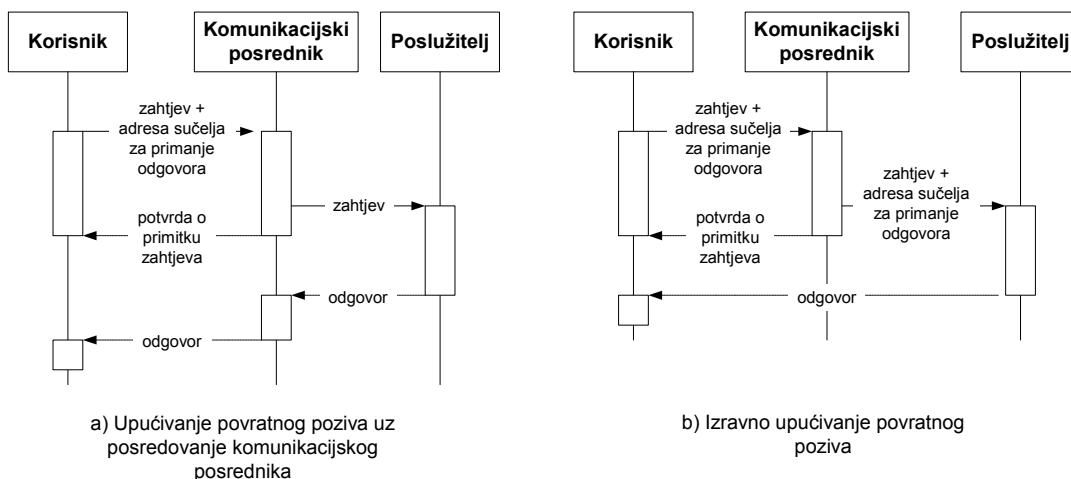


Slika 4.9 Međudjelovanje raspodijeljenih računalnih procesa zasnovano na metodi prozivanja

Proces korisnik poziva uslugu procesa poslužitelja slanjem poruke sa zahtjevom komunikacijskom posredniku. Komunikacijski posrednik sprema poruku u red poruka, a korisniku vraća oznaku spremljene poruke. Primljenu oznaku proces korisnik u nastavku koristi za pribavljanje poruke s rezultatom poziva usluge. Komunikacijski posrednik isporučuje zaprimljenu poruku procesu poslužitelju. Nakon obrade zahtjeva iz poruke, proces poslužitelj oblikuje poruku s rezultatom obrade. Oblikovanu poruku šalje u red poruka. Za vrijeme dok proces poslužitelj obrađuje poruku, proces korisnik pokušava uzastopnim prozivanjem komunikacijskog posrednika pribaviti poruku s rezultatom obrade. Veći broj procesa korisnika i procesa poslužitelja može istodobno komunicirati putem jednog reda poruka. Isto tako, jedan proces korisnik može poslati veći broj poruka sa zahtjevima prije nego što pribavi prvu poruku s odgovorom. Zbog toga je pri svakom prozivanju komunikacijskog posrednika potrebno navesti oznaku poruke sa zahtjevom za koju se traži poruka s odgovorom. Ako je tražena poruka s odgovorom već pristigla u red poruka, ona se isporučuje procesu korisniku. U suprotnom, proces korisnik dobiva poruku kojom ga komunikacijski posrednik obavještava da odgovor pokuša pribaviti kasnije jer poruka s odgovorom od procesa poslužitelja još nije zaprimljena.

4.2.2 Povratni poziv

Razmjena poruka putem redova poruka zasnovana na metodi povratnog poziva (engl. *callback*) [69] također je svojstvena povezivanju procesa među kojima vrijede odnosi korisnika i poslužitelja. Za razliku od metode prozivanja koja koristi mehanizam uzastopnog ispitivanja raspoloživosti poruke u redu poruka, metoda pozivanja usluge s povratnim pozivom zasniva se na obavješćivanju procesa korisnika o prisutnosti poruke s odgovorom procesa poslužitelja. Postupak pozivanja usluge metodom povratnog poziva prikazan je slikom 4.10.



Slika 4.10 Međudjelovanje raspodijeljenih računalnih procesa zasnovano na metodi povratnog poziva

Svaki proces koji s udaljenim procesom nastoji uspostaviti komunikaciju zasnovanu na metodi povratnog poziva izlaže sučelje za prihvati obavijesti o pristigloj poruci s odgovorom procesa poslužitelja. Tijekom slanja poruke sa zahtjevom komunikacijskom posredniku, proces korisnik u poruci navodi vlastitu adresu i komunikacijske parametre sučelja za prihvati obavijesti. Komunikacijski posrednik potvrđuje primitek zahtjeva procesu korisniku te proslijedi zahtjev procesu poslužitelju. Nakon što proces poslužitelj završi s obradom zahtjeva, on započinje postupak vraćanja odgovora procesu korisniku. Vraćanje odgovora u povratnom pozivu moguće je izvesti na dva načina. Slikom 4.10a prikazana je izvedba s upućivanjem povratnog poziva uz posredovanje komunikacijskog posrednika. U tom slučaju proces poslužitelj poruku s odgovorom šalje u red poruka kod komunikacijskog posrednika, a komunikacijski posrednik na osnovi adrese procesa korisnika i komunikacijskih parametara njegova sučelja za prihvati obavijesti proslijedi odgovor procesu korisniku. Slikom 4.10b prikazana je izvedba s izravnim upućivanjem povratnog poziva od procesa poslužitelja prema procesu korisniku. U tom slučaju proces poslužitelj poruku s odgovorom

šalje izravno procesu korisniku. Kako bi doznao adresu sučelja na kojem proces korisnik očekuje primanje odgovora u povratnom pozivu, komunikacijski posrednik procesu poslužitelju, uz zahtjev, prosljeđuje i adresu korisnikova sučelja za primanje odgovora.

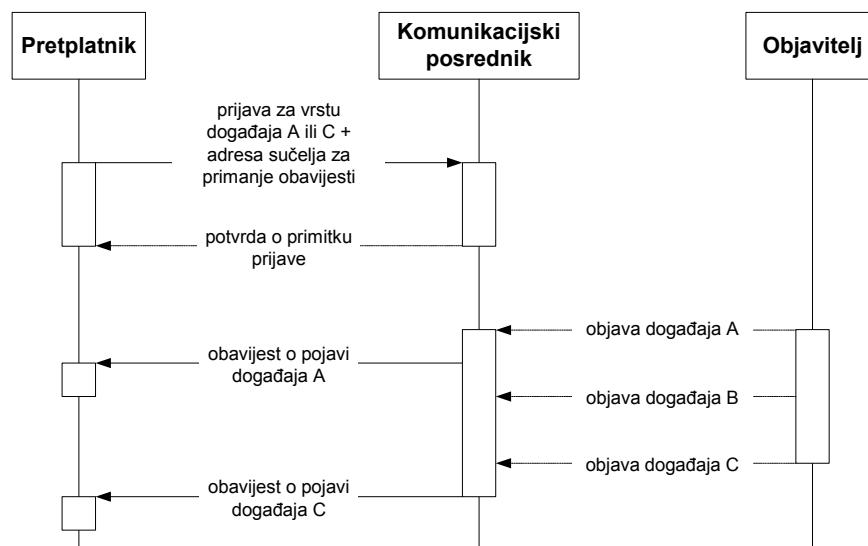
Metodu povratnog poziva moguće je iskoristiti za pozivanje složenih usluga koje se sastoje od slijednog pozivanja više pojedinačnih usluga. Proces korisnik poziva složenu uslugu slanjem poruke sa zahtjevom u red poruka. Poruka sa zahtjevom, uz podatke zahtjeva, sadrži adresu i komunikacijske parametre korisnikova sučelja za prihvat obavijesti o pristiglom odgovoru. Komunikacijski posrednik poruku prosljeđuje procesu poslužitelju prve usluge u nizu koja čini složenu uslugu. Nakon što proces poslužitelj prve usluge završi s obradom poruke, poruka s rezultatima obrade šalje se u red poruka. Kao odredište poruke, trenutni proces poslužitelj navodi adresu sljedećeg procesa poslužitelja u nizu, a u poruku uključuje podatke o adresi i komunikacijskim parametrima sučelja za prihvat obavijesti procesa korisnika. Posljednji proces u nizu oblikuje poruku s konačnim rezultatom obrade te je šalje u red poruka. Komunikacijski posrednik dohvata poruku iz reda poruka i isporučuje je procesu korisniku.

Osim mogućnosti pozivanja složenih usluga, komunikacijski model zasnovan na metodi povratnog poziva pruža mogućnost preusmjeravanja poruke s odgovorom trećem procesu. Proces korisnik tijekom slanja poruke sa zahtjevom u red poruka, umjesto vlastite adrese i komunikacijskih parametara vlastitog sučelja za prihvat obavijesti o pristiglom odgovoru, u poruku uključuje adresu i komunikacijske parametre sučelja za prihvat obavijesti procesa kojemu treba proslijediti odgovor. Poruka s odgovorom poslužitelja je u tom slučaju naslovljena na treći proces te je komunikacijski posrednik, umjesto procesu korisniku, isporučuje trećem procesu.

4.2.3 Objava/preplata

Metode prozivanja i povratnog poziva pogodne su za primjenu u sustavima s jednim potrošačem poruke. Ove dvije metode uglavnom se koriste za dvosmjernu komunikaciju u kojoj proizvođač poruke na poslanu poruku očekuje odgovor od potrošača. Druga komunikacijska paradigma često upotrebljavana u raspodijeljenim računalnim sustavima jest slanje jednosmjerne obavijesti o određenom događaju, odnosno objavljivanje određene informacije. Ovom paradigmom pošiljatelj obavijesti slanjem poruke u red poruka obaviještava unaprijed nepoznat broj primatelja o pojavljivanju događaja. Komunikacijska metoda koja zadovoljava postavljene zahtjeve je metoda objava/preplata [8, 69].

Metoda objava/preplata uvodi uloge procesa objavitelja (engl. *publisher*) i procesa preplatnika (engl. *subscriber*). Proces objavitelj objavljuje obavijesti o pojavljivanju događaja slanjem poruka u red poruka. Proces preplatnik je potrošač objavljenih obavijesti, odnosno poruke kojom se obavijest objavljuje. Komunikacija zasnovana na metodi objava/preplata prikazana je slikom 4.11.



Slika 4.11 Međudjelovanje raspodijeljenih računalnih procesa zasnovano na metodi objava/preplata

Prijavom pri komunikacijskom posredniku, proces preplatnik izražava svoje zanimanje za primanje obavijesti o pojavljivanju određenih vrsta događaja. U trenutku pojavljivanja događaja, proces objavitelj oblikuje poruku koja sadrži obavijest o događaju. Poruku kojom objavljuje pojavu događaja, proces objavitelj šalje u red poruka kod komunikacijskog posrednika. Komunikacijski posrednik pretražuje listu procesa preplatnika koji su izrazili zanimanje o primanju obavijesti za objavljenu vrstu događaja te svim zainteresiranim preplatnicima prosljeđuje primljenu poruku. Za svaku poruku objavljinu od strane procesa objavitelja, komunikacijski posrednik može imati prijavljenog nijednog, jednog ili više procesa preplatnika. Svaki proces preplatnik prima obavijesti o onim događajima za koje je izrazio zanimanje prijavom pri komunikacijskom posredniku.

Ovisno o ostvarenju komunikacijskog posrednika i vrsti objavljenih događaja, zadržavanje poruke o objavljenom događaju u redu poruka može biti različito. Tri najčešće korištene tehnike za izbacivanje poruke iz reda poruka su trenutno izbacivanje, izbacivanje nakon prvog trošenja i stalno zadržavanje. U sustavu s trenutnim izbacivanjem, komunikacijski posrednik u trenutku zaprimanja poruke od procesa objavitelja prosljeđuje poruku svim zainteresiranim preplatnicima. Odmah nakon prosljedivanja, komunikacijski

posrednik briše poruku iz reda poruka. Ako u trenutku zaprimanja poruke nema prijavljenih pretplatnika, nijedan od procesa neće biti obaviješten o pojavi događaja. U sustavu s izbacivanjem nakon prvog trošenja, komunikacijski posrednik u trenutku zaprimanja poruke od procesa objavitelja prosljeđuje poruku svim zainteresiranim pretplatnicima, a potom je briše. Ako u trenutku zaprimanja poruke nema prijavljenih pretplatnika, komunikacijski posrednik zadržava poruku u redu poruka sve do pojave prvog pretplatnika za objavljenu vrstu događaja. Nakon isporuke poruke prvom pretplatniku, poruka se briše iz reda poruka. U sustavu sa stalnim zadržavanjem, poruke se ne brišu iz reda poruka. U trenutku objavljivanja događaja, svi pretplatnici se obavještavaju o pojavi događaja. Ako u nekom kasnijem trenutku novi proces izrazi zanimanje za objavljenu vrstu događaja, on je u mogućnosti primiti sve prethodne poruke koje se odnose na tu vrstu događaja.

4.2.4 Analiza primjenjivosti metoda razmjene poruka

Tri osnovne metode asinkrone komunikacije zasnovane na razmjeni poruka imaju svojih posebnosti glede područja primjene i programskog ostvarenja. Područje primjene raspodijeljenog računalnog sustava uglavnom uvjetuje i metodu komunikacijskog međudjelovanja koju je potrebno primjeniti u danim okolnostima. Ako se komunikacijske veze uspostavljaju između pojedinih parova raspodijeljenih procesa, onda je potrebno primijeniti komunikacijski model s jednim potrošačem poruke uz uporabu metode prozivanja ili metode povratnog poziva. Ako se komunikacijske veze uspostavljaju između skupine raspodijeljenih procesa, onda je potrebno primjeniti komunikacijski model s više potrošača poruke zasnovan na objavljivanju obavijesti o pojavljivanju događaja uz uporabu metode objava/pretplata.

Izbor metode razmjene poruke među raspodijeljenim računalnim procesima utječe na složenost programskog ostvarenja i učinkovitost raspodijeljenog sustava zasnovanog na razmjeni poruka. Jednostavnost programske izvedbe svojstvena je metodi prozivanja. Međutim, zbog učestalog ispitivanja komunikacijskog posrednika o raspoloživosti očekivane poruke, ta metoda pokazuje manjkavosti glede korištenja komunikacijskih veza i razmernog rasta raspodijeljenog sustava. S druge strane, učinkovitije korištenje komunikacijskih veza i smanjeno opterećenje komunikacijskog posrednika svojstveno metodi povratnog poziva iziskuje složeniju programsku izvedbu komunikacijskog posrednika i procesa korisnika usluge. Metoda objava/preplata je najnapredniji način razmjene poruka među raspodijeljenim procesima. Programsко ostvarenje raspodijeljenog sustava zasnovanog na metodi objava/preplata znatno je složenije od metode prozivanja i povratnog poziva, a učinkovitost ovisi o programskoj izvedbi mehanizama za povezivanje objavljenih događaja s

procesima preplatnicima. Tablicom 4.1 prikazane su ključne razlike u područjima primjene, načinima programskog ostvarenja te prednostima i nedostacima raspodijeljenih sustava zasnovanih na komunikacijskim metodama prozivanja, povratnog poziva i objava/preplata.

Tablica 4.1 Usporedba područja primjene, složenosti programskog ostvarenja te prednosti i nedostataka metoda razmjenе poruka

Komunikacijska metoda	Područje primjene	Složenost programskog ostvarenja	Prednosti i nedostaci
Prozivanje	Raspodijeljeni sustavi s relativno malim brojem raspodijeljenih procesa, sustavi s predvidljivim vremenom odziva	Najjednostavnija od svih triju metoda razmjenе poruka	Visok stupanj opterećenja komunikacijskih veza i komunikacijskog posrednika zbog učestalog ispitivanja raspoloživosti poruke, ograničena potpora razmijernom rastu raspodijeljenog sustava s obzirom na broj raspodijeljenih procesa
Povratni poziv	Sustavi s dugotrajnim vremenom odziva, usluge koje iziskuju posredovanje čovjeka	Složenija programska izvedba, proces korisnik mora izložiti sučelje za prihvat obavijesti o raspoloživosti poruke	Niski stupanj opterećenja komunikacijskih veza i komunikacijskog posrednika, dobra potpora razmijernom rastu raspodijeljenog sustava s obzirom na broj raspodijeljenih procesa
Objava/preplata	Sustavi i usluge elektroničkog obavješćivanja, mreže osjetila (engl. <i>sensor networks</i>)	Složena programska izvedba, potrebno je voditi listu mogućih događaja i listu preplatnika za pojedine vrste događaja, komunikacijski posrednik mora obavljati postupak povezivanja događaja s procesima preplatnicima, svaki preplatnik mora izložiti sučelje za prihvat obavijesti o objavljenom događaju	Učinkovitost sustava znatno ovisi o ostvarenju mehanizama za povezivanje događaja s procesima preplatnicima (povezivanje unutar posrednika opterećuje posrednika, povezivanje izvan posrednika opterećuje komunikacijske veze)

5 Prividna raspodijeljena računalna okolina

Mogućnosti iskorištavanja globalne računalne mreže Internet u gospodarstvu, poslovanju, upravi, zabavi i školovanju u današnje su vrijeme prepoznate od strane krajnjih korisnika. Razlog tome je prije svega *World Wide Web* usluga koja je svojom pojavom prije nešto više od desetak godina uporabu mreže Internet približila običnim korisnicima osobnih računala. *World Wide Web* usluga prvobitno je osmišljena kao sustav za objavljivanje i razmjenu informacija u elektroničkom obliku među članovima znanstvene zajednice, koristeći pritom komunikacijsku infrastrukturu mreže Internet. *World Wide Web* usluga ubrzo je nadišla okvire znanstvene zajednice te je postala osnovno sredstvo za objavljivanje, pristup i razmjenu različitih vrsta informacija u najširim slojevima društva.

U njenom današnjem obliku većina krajnjih korisnika doživljava mrežu Internet i *World Wide Web* uslugu kao globalni raspodijeljeni sustav za objavljivanje, pretraživanje i razmjenu najrazličitijih vrsta informacija. Krajnji korisnici mreže Internet imaju vrlo malo ili uopće nemaju mogućnosti nadzora nad ponašanjem mreže, načinima njezine uporabe i iskorištavanja njezinih mogućnosti te sadržaja koji je tom mrežom dostupan. Sve što krajnjem korisniku mreže Internet stoji na raspolaganju svodi se na objavljivanje vlastitog sadržaja koji na taj način postaje javno dostupan, odnosno na pristupanje i korištenje sadržaja koji su drugi korisnici učinili javno dostupnim. Dodatni nedostatak trenutnog stanja mreže Internet predstavljaju primjenske usluge uglavnom namijenjene uporabi koja zahtijeva izravno međudjelovanje čovjeka i računalnog sustava. Suvremeno informacijsko društvo zahtijeva globalnu svjetsku mrežu koja pruža potporu automatskoj uporabi raspoloživih sredstava bez potrebe za posredovanjem čovjeka te visok stupanj prilagodljivosti potrebama pojedine društvene skupine ili pojedinca.

Primjena informacijskih tehnologija u suvremenim sveprisutnim i prožimajućim računalnim sustavima (engl. *ubiquitous and pervasive computing*) te sustavima za svakodnevnu pomoć u obavljanju uobičajenih poslova zasnovanih na načelima neprimjetnog računarstva (engl. *invisible computing*) zahtijeva mogućnost prilagodbe mreže Internet kroz mogućnost njezina programiranja i programske upravljanja njezinim sredstvima. Istraživačka grupa *RIS* (engl. *Research on Internet/Intranet Systems*) okupljena na *Zavodu za elektroniku, mikroelektroniku računalne i inteligentne sustave* na *Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu* razvila je prototip raspodijeljenog posredničkog sustava (engl. *application-level middleware*) koji omogućava prilagodbu mreže Internet pojedinim primjenskim potrebama, nazvavši je prividnom raspodijeljenom računalnom okolinom (engl. *Virtual Distributed Environment*). Prividna raspodijeljena računalna okolina razvijena je u

suradnji s *Institutom za telekomunikacije* tvrtke *Ericsson Nikola Tesla d.d.* iz Zagreba. Rezultati istraživačkog rada iskorišteni su kao osnovica za pokretanje tehnologiskog projekta *CroGrid* pod pokroviteljstvom *Ministarstva znanosti, obrazovanja i športa Republike Hrvatske*.

Prividna raspodijeljena računalna okolina je raspodijeljeni računalni sustav koji pruža potporu za oblikovanje, razvoj, postavljanje i izvođenje raspodijeljenih programskih sustava i poslovnih procesa zasnovanih na uslugama. Uporabom prividne raspodijeljene računalne okoline, prostor globalne mreže Internet pretvara se iz globalnog informacijskog sustava u globalni raspodijeljeni programirljivi računalni sustav.

5.1 Programski model zasnovan na uslugama

Bilo koji programirljivi sustav, bio on jedinstvena cjelina ili sastavljen od raspodijeljenih elemenata, zahtijeva primjenu određenog programskog modela, odnosno programske paradigme koja se koristi za izgradnju programskih cjelina. Programske cjeline koje se grade i izvode unutar prividne raspodijeljene računalne okoline su raspodijeljeni programski sustavi zasnovani na uslugama (engl. *service-oriented distributed applications*). Za izgradnju raspodijeljenih programskih sustava zasnovanih na uslugama koristi se programski model zasnovan na uslugama (engl. *Service-Oriented Programming Model*) [89].

Osnovni gradivni elementi pri oblikovanju i izgradnji raspodijeljenih programskih sustava uporabom programskog modela zasnovanog na uslugama su programske usluge (engl. *services*). Primjena programskog modela zasnovanog na uslugama koji koristi prividna raspodijeljena računalna okolina razlikuje tri vrste programskih usluga: primjenske usluge (engl. *application services*), usluge za suradnju i natjecanje (engl. *coopetition services*, *coopetition* = *cooperation* + *competition*) te raspodijeljene programe (engl. *distributed programs*).

Primjenske usluge ostvaruju funkcionalnosti svojstvene pojedinim područjima primjene. Primjerice, primjenske usluge mogu pružati funkcionalnosti izračunavanja prostih faktora velikih brojeva, davati pregled stanja burzovnih izvješća s vrijednostima dionica, obavljati rezervaciju zrakoplovnih karata, nuditi mogućnost obavljanja bankovnih transakcija i slično.

Raspodijeljeni programi su složene programske usluge (engl. *composite services*) koje omogućavaju povezivanje određenog broja pojedinačnih primjenskih usluga u nove i složenije primjenske usluge. Raspodijeljeni programi sadrže programsku logiku potrebnu za

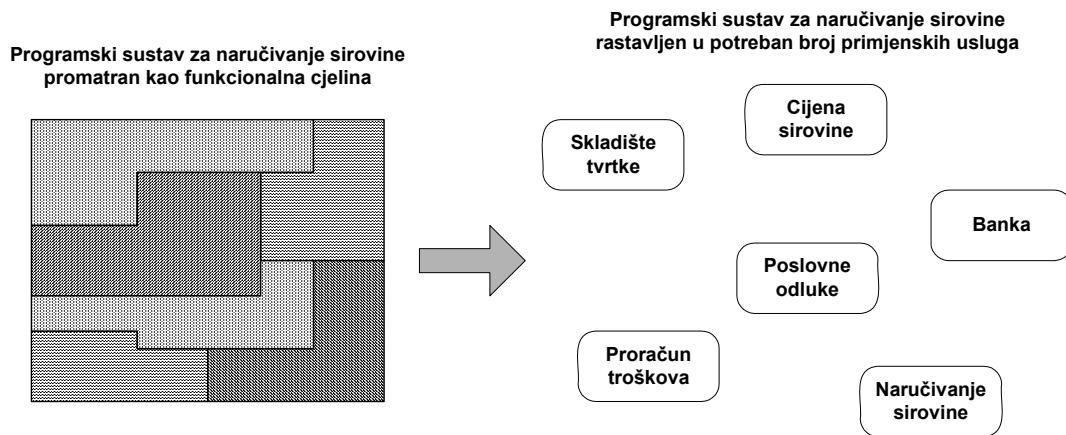
usklađivanjem rada primjenskih usluga uporabom usluga za suradnju i natjecanje. Programska logika svojstvena području primjene raspodijeljenog programskog sustava izdvojena je iz raspodijeljenih programa i u potpunosti je ostvarena u primjenskim uslugama raspodijeljenim diljem mreže Internet. Raspodijeljeni programi sadrže logiku za usklađivanje rada primjenskih usluga. Tijekom izvođenja, raspodijeljeni programi poprimaju oblik primjenske usluge, povezujući na taj način jednostavne primjenske usluge ili druge raspodijeljene programe u složenije primjenske usluge.

Povezivanje primjenskih usluga u raspodijeljene programe te omogućavanje njihove međusobne komunikacije i usklađivanje rada (engl. *synchronization*) postiže se primjenom usluga za suradnju i natjecanje [89, 90]. Prividna raspodijeljena računalna okolina koristi četiri vrste usluga za suradnju i natjecanje: binarni semafor, opći semafor, poštanski pretinac i usmjernik događaja. Binarni i opći semafor koriste se za međusobno isključivanje raspodijeljenih programa pri pokušaju istovremenog pristupa do zajedničkih sredstava do kojih pristup treba biti ograničen [91]. Pritom se binarnim semaforom dozvoljava istovremeni pristup do zajedničkog sredstva isključivo jednom raspodijeljenom programu, dok se općim semaforom istovremeni pristup do zajedničkog sredstva dozvoljava većem, ali ograničenom broju raspodijeljenih programa. Poštanski pretinac se, osim za usklađivanje rada raspodijeljenih programa, koristi i za njihovu međusobnu komunikaciju [91]. Poštanski pretinac predstavlja red poruka u koji raspodijeljeni programi upisuju poruke ili ih iz njega čitaju. Čitanje poruke iz poštanskog pretinca je blokirajuća operacija koja uzrokuje zaustavljanje izvođenja raspodijeljenog programa koji pokušava čitati poruku iz praznog pretinca. Izvođenje raspodijeljenog programa je zaustavljeno do nailaska prve poruke koju u pretinac upisuje neki drugi raspodijeljeni program. Razmjenom poruka putem mehanizma poštanskog pretinca ostvaruje se komunikacija među raspodijeljenim programima, dok je blokirajućom operacijom čitanja poštanskog pretinca osigurano usklađivanje njihova rada. Usmjernik događaja se, kao najnaprednija vrsta usluge za suradnju i natjecanje, koristi za ostvarivanje mreža zasnovanih na sadržaju (engl. *content routing and filtering*) [92]. Usmjernik događaja za komunikaciju i usklađivanje rada raspodijeljenih programa koristi mehanizam objava/preplata (engl. *publish/subscribe*). Raspodijeljeni programi prijavljivanjem pri usmjerniku događaja iskazuju zanimanje za određenu vrstu događaja koji se mogu pojaviti u raspodijeljenoj okolini. U postupku prijave, raspodijeljeni programi naznačavaju popis događaja za koje iskazuju zanimanje. Raspodijeljeni programi koji otkrivaju pojavu određenih vrsta događaja dojavljuju pojave događaja usmjerniku događaja. S ciljem uparivanja objavljenih događaja s iskazanim zanimanjima za događaje, usmjernik događaja poziva tumač događaja. Ako tumač događaja uspije spojiti objavljeni događaj s

iskazanim zanimanjem za pojavu događaja, usmjernik događaja dojavljuje pojavu događaja raspodijeljenom programu koji je iskazao zanimanje za dotični događaj.

5.1.1 Faza oblikovanja raspodijeljenog programskog sustava

Razvoj raspodijeljenih programskih sustava zasnovanih na uslugama za izvođenje u prividnoj raspodijeljenoj računalnoj okolini sastoji se od faze oblikovanja i faze programskog ostvarenja. Oblikovanje raspodijeljenih programskih sustava započinje rastavljanjem programske logike raspodijeljenog programskog sustava u potreban broj primjenskih usluga. Primjerice, neka uporabom programskog modela zasnovanog na uslugama proizvodna tvrtka želi ostvariti raspodijeljeni programski sustav za automatizirano naručivanje sirovina od dobavljača. Analizom načina poslovanja tvrtke i postupka naručivanja sirovina, programsku logiku raspodijeljenog promjenskog programa moguće je rastaviti na šest primjenskih usluga: uslugu skladišta za pregled stanja sirovina na skladištu tvrtke, uslugu dobavljača za pregled trenutnog stanja cijene sirovine, uslugu banke za uvid u finansijsko stanje bankovnih računa tvrtke, uslugu tvrtke za izračunavanje potrebne količine sirovine i ukupnog troška narudžbe, uslugu poslovnih odluka tvrtke za određivanje strategije poslovanja i usklađivanje prihoda i rashoda te uslugu dobavljača za naručivanje sirovine.

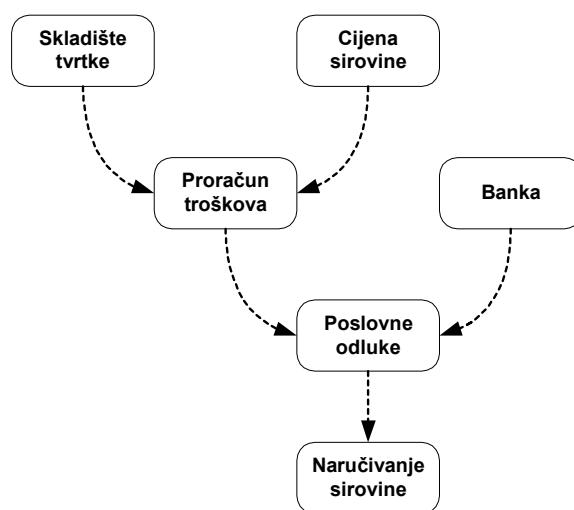


Slika 5.1 Rastavljanje programske logike raspodijeljenog programskog sustava u skup primjenskih usluga

Slikom 5.1 prikazano je rastavljanje programske logike raspodijeljenog programskog sustava za automatizirano naručivanje sirovine u skup primjenskih usluga utvrđen analizom poslovnog procesa. Primjenske usluge raspodijeljene su diljem mreže Internet. U primjeru na slici 5.1, usluge skladišta, poslovnih odluka i proračuna troškova smještene su i održavane od same tvrtke, banka je usluga smještena i održavana od strane poslovnice banke putem

koje tvrtka obavlja poslovanje, dok su usluge cijene sirovine i naručivanja sirovina pod nadzorom dobavljača sirovine.

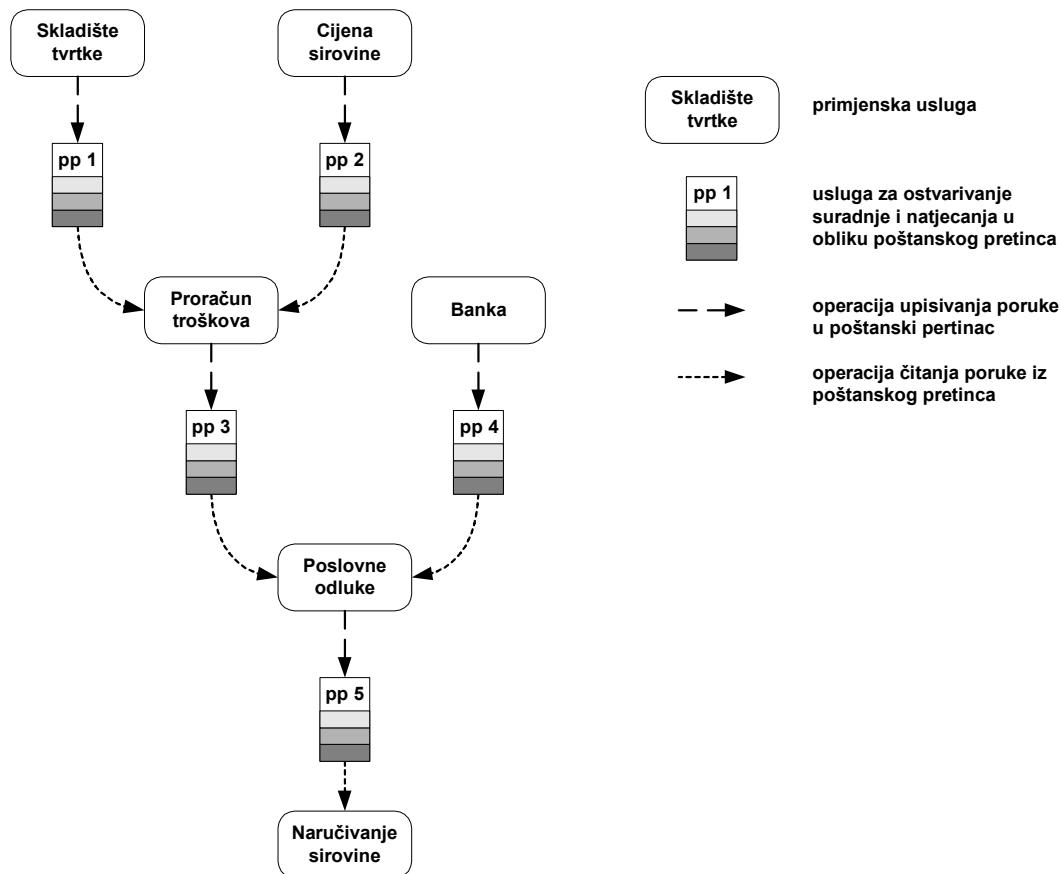
Nakon što su utvrđeni elementarni dijelovi raspodijeljenog programskog sustava u obliku primjenskih usluga rasprostranjenih diljem mreže Internet, potrebno je pristupiti definiranju logike za usklađivanje njihova rada. Definiranje logike za usklađivanje rada primjenskih usluga obuhvaća organizaciju pozivanja primjenskih usluga u redoslijed određen područjem primjene te definiranje mjesta i načina njihova usklađivanja (engl. *synchronization points*). Slikom 5.2 prikazan je redoslijed pozivanja primjenskih usluga s prikazanim točkama usklađivanja njihova rada.



Slika 5.2 Definiranje logike za usklađivanje rada primjenskih usluga pomoću redoslijeda njihova pozivanja i točaka usklađivanja njihova rada

Na početku izvođenja raspodijeljenog programskog sustava zasnovanog na uslugama pozivaju se usluge za pregled stanja sirovine na skladištu tvrtke i usluge za pribavljanje trenutne cijene sirovine od dobavljača. Izvođenje tih dviju usluga u ovom je slučaju neovisno jedno o drugom te ih je moguće pozvati i izvoditi istodobno. Rezultati pribavljeni pozivanjem usluge skladišta i usluge cijene sirovine koriste se kao ulazni parametri poziva usluge proračuna troškova. Na osnovi ulaznih parametara, usluga proračuna troškova određuje količinu sirovine koju je potrebno naručiti te izračunava ukupni trošak narudžbe. Budući da usluga proračuna troškova treba rezultate izvođenja usluge skladišta i usluge cijene sirovine, njezino izvođenje može započeti tek po završenom izvođenju obje prethodne usluge. Usluga poraćuna troškova ima definirane dvije točke za usklađivanje rada s ostalim uslugama, jednu s uslugom skladišta, a drugu s uslugom cijene sirovine. Istovremeno s izvođenjem usluge proračuna troškova narudžbe, usluži banke je moguće poslati upit s

ciljem pribavljanja trenutnog financijskog stanja na bankovnom računu tvrtke. Budući da je izvođenje usluge banke također neovisno o svim drugim uslugama, njezino je izvođenje moglo započeti i ranije, istovremeno s izvođenjem usluge skladišta i usluge cijene sirovine. Rezultate proračuna troškova i financijskog stanja bankovnog računa koristi usluga za potporu poslovnom odlučivanju, kako bi na osnovi zahtijevane količine sirovina i raspoloživih financijskih sredstava uskladila zahjeve i mogućnosti narudžbe. Usluga za donošenje poslovnih odluka, stoga, ima definirane točke usklađivanja s uslugom proračuna troškova i uslugom banke. Na osnovi rezultata izvođenja usluge za donošenje poslovnih odluka, poziva se usluga za naručivanje sirovine od dobavljača. Budući da za početak njezina izvođenja prethodno treba završiti izvođenje svih ostalih usluga, njezino je izvođenje potrebno uskladiti s izvođenjem tih usluga. To se u prikazanom slučaju postiže definiranjem točke usklađivanja s uslugom za donošenje poslovnih odluka.



Slika 5.3 Korištenje usluge poštanskog pretinca za ostvarivanje komunikacije i usklađivanje rada primjenskih usluga

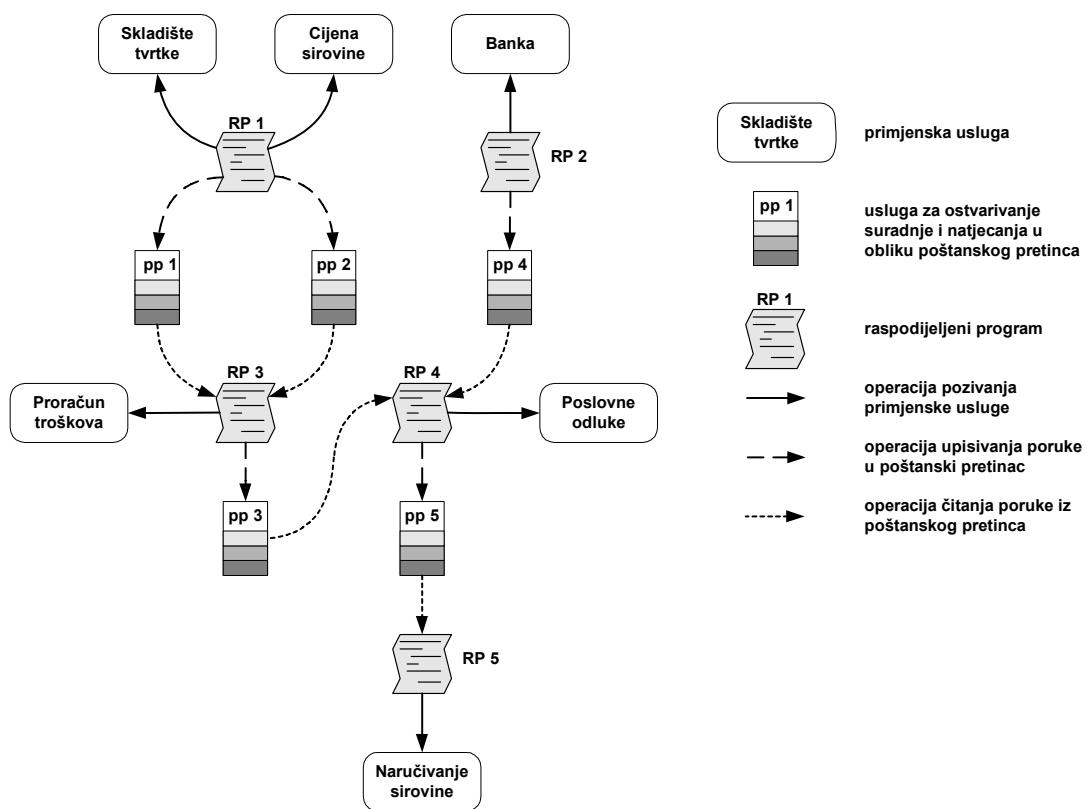
Na osnovi logike za usklađivanje rada primjenskih usluga utvrđuje se vrsta i potreban broj usluga za suradnju i natjecanje primjenskih usluga. Budući da u razmatranom primjeru

raspodijeljenog programskog sustava nema potrebe za međusobnim isključivanjem izvođenja primjenskih usluga pri korištenju ograničenog sredstva, usluge binarnog i općeg semafora nije potrebno koristiti. Isto tako nije potrebno upotrijebiti ni uslugu usmjernika događaja, budući da nijedna od korištenih primjenskih usluga ne koristi model komunikacije zasnovan na događajima. Usluge ostvaruju međusobnu komunikaciju proslijđivanjem rezultata izvođenja jedne usluge kao ulaznih parametara drugoj usluzi. Osim toga, usluge za čije su izvođenje potrebni ulazni parametri ne smiju započeti s izvođenjem prije nego što im ti parametri postanu dostupni. Komunikaciju i usklađivanje rada primjenskih usluga moguće je osigurati korištenjem usluga za suradnju i natjecanje u obliku poštanskih pretinaca. Korištenje usluge poštanskog pretinca za ostvarivanje komunikacije i usklađivanje rada primjenskih usluga prikazano je slikom 5.3. Za ostvarivanje komunikacije i suradnje primjenskih usluga zadanog raspodijeljenog programskog sustava zasnovanog na uslugama potrebno je koristiti pet poštanskih pretinaca.

Programska logika za usklađivanje rada primjenskih usluga rastavlja se u određen broj raspodijeljenih programa. Svaki raspodijeljeni program je nezavisni programski element koji za vrijeme izvođenja poprima obilježja složene primjenske usluge. Jednim raspodijeljenim programom ostvaruje se dio logike za usklađivanje rada primjenskih usluga, dok je svim raspodijeljenim programima koji pripadaju jednom raspodijeljenom programskom sustavu zasnovanom na uslugama ta logika ostvarena u cjelini. Logika za usklađivanje rada primjenskih usluga ostvaruje se pozivanjem primjenskih usluga i usluga za suradnju i natjecanje. Pozivanjem primjenskih usluga, raspodijeljeni program obavlja računalne operacije svojstvene području primjene raspodijeljenog programskog sustava. Pozivanjem usluga za suradnju i natjecanje, raspodijeljeni program usklađuje izvođenje primjenskih usluga. Redoslijed kojim raspodijeljeni program poziva primjenske usluge i usluge za suradnju i natjecanje određen je radnim tijekom (engl. *workflow*) dijela logike raspodijeljenog programskog sustava ostvarenog tim raspodijeljenim programom.

Slikom 5.4 prikazana je raspodjela logike za usklađivanje rada primjenskih usluga za automatizirano naručivanje sirovina u obliku pet raspodijeljenih programa. Svaki od pet raspodijeljenih programa ostvaruje dio ukupne logike za usklađivanje rada primjenskih usluga. Svi raspodijeljeni programi koji pripadaju istom raspodijeljenom programskom sustavu zasnovanom na uslugama istodobno započinju s izvođenjem. Daljnje usklađivanje njihova rada obavlja se pozivima usluga za suradnju i natjecanje. Raspodijeljeni programi *RP 1* i *RP 2* na početku izvođenja pozivaju primjenske usluge, a izvođenje završavaju upisivanjem poruka u poštanske pretince. Ta se dva programa mogu nesmetano obaviti istodobno. S druge strane, raspodijeljeni programi *RP 3* i *RP 4* na početku izvođenja čitaju

poruke s poštanskih pretinaca, a tek potom pozivaju primjenske usluge. Budući da je operacija čitanja praznog poštanskog pretinca blokirajuća, ti programi privremeno zaustavljaju svoje izvođenje. Izvođenje raspodijeljenog programa *RP 3* moguće je nastaviti u trenutku kada raspodijeljeni program *RP 1* upiše poruke u poštanske pretince *pp 1* i *pp 2*. Raspodijeljeni program *RP 4* nastavlja s izvođenjem nakon što raspodijeljeni program *RP 2* upiše poruku u poštanski pretinac *pp 4* te nakon što raspodijeljeni program *RP 3* upiše poruku u poštanski pretinac *pp 3*. Istovjetno razmatranje vrijedi za raspodijeljeni program *RP 5*. Njegovo je izvođenje privremeno zaustavljeno do trenutka dok mu raspodijeljeni program *RP 4* ne upiše poruku u poštanski pretinac *pp 5*. Rastavljanjem logike za usklađivanje rada primjenskih usluga u raspodijeljene programe završava faza oblikovanja raspodijeljenog programskog sustava zasnovanog na uslugama.



Slika 5.4 Raspodijeljivanje logike za usklađivanje rada primjenskih usluga u skup raspodijeljenih programa

5.1.2 Faza programskog ostvarenja raspodijeljenog programskog sustava

Programsko ostvarenje raspodijeljenog programskog sustava zasnovanog na uslugama sastoji se od programskog ostvarenja logike za usklađivanje rada primjenskih usluga u obliku raspodijeljenih programa. Ostvarenje programske logike sastoji se od naredbi za

pozivanje primjenskih usluga te naredbi za pozivanje usluga za suradnju i natjecanje. Za definiranje naredbi raspodijeljenih programa osmišljen je programski jezik *SSCL* (engl. *Simple Service Composition Language*) [93]. Jezik *SSCL* je jednostavni programski jezik za povezivanje programskih usluga.

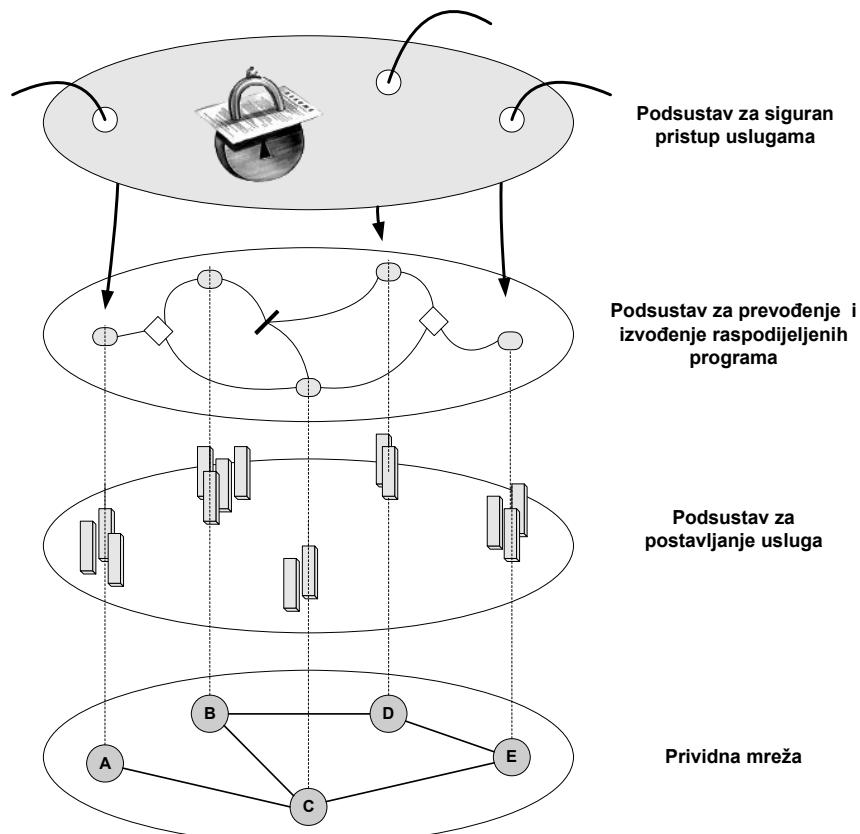
Nakon faze programskog ostvarenja, slijede faza prevodenja raspodijeljenih programa u složene primjenske usluge te faza izvođenja raspodijeljenih programa u obliku složenih primjenskih usluga. Tijekom faze prevodenja, pojednostavljeni i krajnjem korisniku prilagođeni *SSCL* jezik prevodi se u jezik *CL* (engl. *Coopetition Language*) [94, 95]. Jezik *CL* zasnovan je na jeziku *BPEL4WS* (engl. *Business Process Execution Language for Web Services*) [96], koji je standardom propisani jezik za povezivanje programskih usluga. Tijekom izvođenja, jezik *CL* se kroz nekoliko koraka prevodenja prevodi u izvodivi programski kod složene primjenske usluge, koji se zatim tumači i izvodi. Prevodenje raspodijeljenih programa u složene primjenske usluge podrobnije je opisano u [93], dok je njihovo izvođenje objašnjeno u [94, 95].

5.2 Arhitektura prividne raspodijeljene računalne okoline

Prividna raspodijeljena računalna okolina osigurava mehanizme za stvaranje i prilagodbu radnog okruženja mreže Internet potrebama pojedinih raspodijeljenih programskih sustava zasnovanih na uslugama. Ti mehanizmi krajnjem korisniku osiguravaju potporu za razvoj i izvođenje raspodijeljenih programskih sustava, koristeći programski model zasnovan na uslugama. Prividna raspodijeljena računalna okolina sastoji se od prividne mreže, podsustava za postavljanje usluga, podsustava za prevođenje i izvođenje raspodijeljenih programa te podsustava za siguran pristup uslugama. Logička organizacija prividne raspodijeljene računalne okoline prikazana je slikom 5.5. Za programsko ostvarenje svih četiriju podsustava korištena su načela računarstva zasnovanog na uslugama. Sve su sastavne cjeline izgradene od elemenata koji se prema drugim cjelinama ponašaju kao nezavisne programske usluge. Prividna raspodijeljena računalna okolina je, stoga, raspodijeljeni sustav zasnovan na uslugama koji pruža potporu za razvoj i izvođenje raspodijeljenih sustava zasnovanih na uslugama. Za njezinu su izgradnju korištena ista načela izgradnje programske potpore kao što se koriste i pri izgradnji raspodijeljenih programskih sustava pomoću nje same.

Složene programske usluge koje nastaju povezivanjem elementarnih primjenskih usluga i usluga za suradnju i natjecanje u raspodijeljene programe u komunikacijskom prostoru prividne raspodijeljene računalne okoline moguće je raspodjeljivati na dva načina.

Prvi je način uobičajeni postupak raspodjele usluga u komunikacijskom prostoru *World Wide Web* usluge, pri čemu je svaka usluga jednoznačno određena *IP* adresom i jedinstvenom oznakom sredstva (engl. *Uniform Resource Locator – URL*). Taj se postupak naziva raspodjelom primjenskih usluga u fizičkom komunikacijskom prostoru. Međutim, raspodjela usluga u fizičkom komunikacijskom prostoru mreže Internet čvrsto povezuje uslugu s njezinim trenutnim položajem u fizičkoj mreži. Radi olakšane raspodjele usluga i olakšanog programskog ostvarenja raspodijeljenih programa, prividna raspodijeljena računalna okolina nudi mogućnost raspodjele usluga u logičkom komunikacijskom prostoru prividne mreže. Prividna mreža dodjeljuje mrežnim čvorovima logička imena koja su neovisna o njihovim adresama u fizičkoj mreži. Korištenjem komunikacijskog prostora prividne mreže, raspodijeljeni programi su otporni na promjene fizičkih adresa usluga nastalih kao posljedica premeštanja usluge s jednog fizičkog čvora na drugi. Osim otpornosti na promjene u fizičkim adresama, olakšana je i provedba mehanizama otpornosti raspodijeljenih programa na kvarove mrežnih čvorova. Iscrplni opis funkcionalnosti, arhitekture i programskog ostvarenja prividne mreže može se pronaći u nastavku ovog rada, u poglavljima 6, 7 i 8.



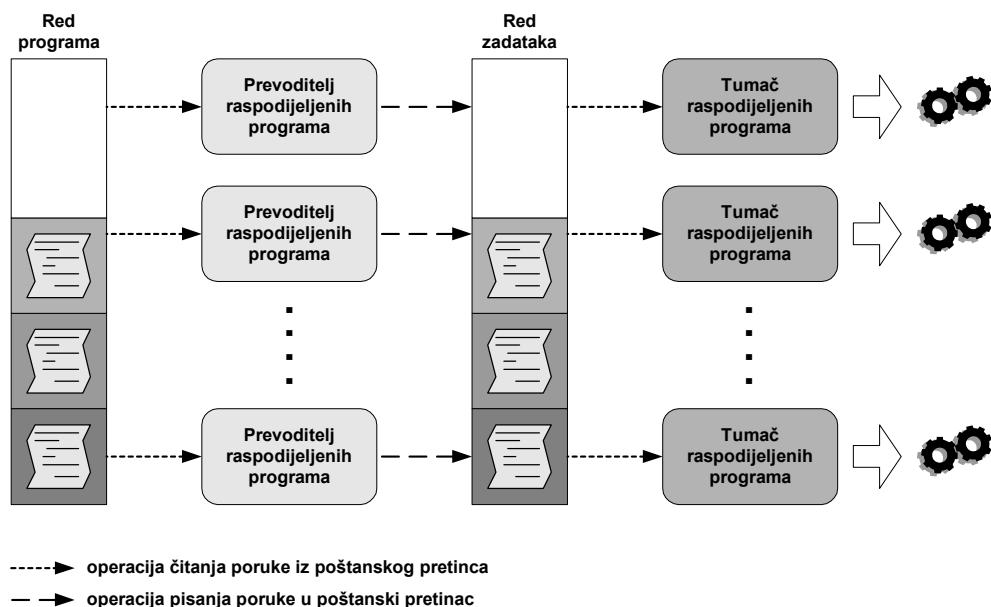
Slika 5.5 Prividna raspodijeljena računalna okolina

Prividna raspodijeljena računalna okolina omogućava postavljanje programskih usluga (engl. *service deployment*) po mrežnim čvorovima te okoline. Postavljanje usluge moguće je izvršiti u fizičkom komunikacijskom prostoru mreže Internet ili u logičkom komunikacijskom prostoru prividne mreže. Postupak postavljanja usluga po čvorovima prividne raspodijeljene računalne okoline poduprt je s tri usluge posebne namjene. Kako bi postavljanje usluge bilo moguće, njezine datoteke za postavljanje moraju biti raspoložive u spremištu usluga (engl. *service repository*). Spremiste usluga je raspodijeljena usluga iz koje se tijekom postupka postavljanja usluge dohvaćaju potrebne datoteke za postavljanje. Ako prije početka postupka postavljanja usluge datoteke za njezino postavljanje nisu raspoložive u spremištu usluga, korisnik mora pokrenuti postupak njihova spremanja u spremište. Za svaku spremljenu uslugu, spremiste usluge sadrži datoteke s izvodivim programskim kôdom usluge, datoteku s opisom usluge u obliku *WSDL* dokumenta te datoteku s uputama za postavljanje usluge. Postupak postavljanja usluge obavlja se slanjem zahtjeva za postavljanjem usluge rasporedioca usluga (engl. *service dispatcher*). Rasporedioca usluga je usluga posebne namjene raspodijeljena po svim čvorovima prividne raspodijeljene računalne okoline. Po zaprimljenom zahtjevu za postavljanjem usluge, rasporedioca usluga dohvaća datoteke za postavljanje usluge iz spremišta usluga te obavlja postupak njezina postavljanja na lokalnom čvoru. Po završenom postupku postavljanja usluge, rasporedioca usluge upisuje podatke o postavljenoj usluzi u imenik usluga (engl. *service registry*). Imenik usluga je usluga posebne namjene koja sadrži podatke o postavljenim uslugama po čvorovima prividne raspodijeljene računalne okoline te njihovim fizičkim ili logičkim adresama. Iscrpni opis funkcionalnosti, arhitekture i programskog ostvarenja podsustava za postavljanje usluga može se pronaći u [97, 98].

Povezivanje elementarnih usluga u složene primjenske usluge u obliku raspodijeljenih programa, kao i izvođenje raspodijeljenih programa omogućeno je mehanizmima podsustava za prevodenje i izvođenje raspodijeljenih programa. Izvršavanje složene usluge u obliku raspodijeljenog programa sastoji se od dva koraka. U prvom koraku se raspodijeljeni program napisan u jeziku *SSCL* prevodi u jezik *CL*. Raspodijeljeni program zapisan sintaksom jezika *SSCL* naziva se programom (engl. *program*), dok se raspodijeljeni program zapisan sintaksom jezika *CL* naziva zadatkom (engl. *job*). Prevodenje raspodijeljenih programa iz jezika *SSCL* u jezik *CL* obavlja usluga posebne namjene nazvana prevoditeljem raspodijeljenih programa (engl. *Distributed Program Translator*). Drugi korak izvršavanja složene usluge započinje tumačenjem naredbi jezika *CL*, njihovim prevodenjem u izvodivi programski kôd te neposrednim izvodenjem. Tumačenje i izvodenje naredaba jezika *CL*

obavlja usluga posebne namjene pod nazivom tumač raspodijeljenih programa (engl. *Distributed Program Interpreter*).

Budući da je raspodijeljeni programski sustav zasnovan na uslugama sastavljen od većeg broja raspodijeljenih programa, za njegovo je izvođenje potrebno ostvariti sustav za raspoređivanje raspodijeljenih programa. U trenutnoj inačici, prividna raspodijeljena računalna okolina za raspoređivanje raspodijeljenih programa koristi jednostavan mehanizam zasnovan na redu programa (engl. *program queue*) i redu zadataka (engl. *job queue*). Sustav za raspoređivanje raspodijeljenih programa prikazan je slikom 5.6. Svi raspodijeljeni programi koji pripadaju zadanom raspodijeljenom programskom sustavu na početku izvođenja pošalju se u red programa. Određen broj prevoditelja raspodijeljenih programa dohvaća raspodijeljene programe iz reda programa, prevodi ih u zadatke te ih upisuje u red zadataka. Iz reda zadataka dohvaća ih određeni broj tumača raspodijeljenih programa, tumači ih i izvodi. Za ostvarenje reda programa i reda zadataka pogodnom se pokazala usluga za suradnju i natjecanje u obliku poštanskog pretinca. Iscrpni opis funkcionalnosti, arhitekture i programske ostvarenja podsustava za prevođenje i izvođenje raspodijeljenih programa može se pronaći u [93, 94, 95].



Slika 5.6 Sustav za raspoređivanje raspodijeljenih programa prividne raspodijeljene računalne okoline

Pristup do mehanizama prividne raspodijeljene računalne okoline te elementarnih i složenih usluga koje se njome izgrađuju je pod strogim nadzorom. Budući da je korisnicima sustava omogućeno postavljanje i izvođenje usluga u raspodijeljenoj okolini, u sustav se neizbjegno unose znatni sigurnosni rizici. Komunikacija među raspodijeljenim dijelovima

sustava treba biti zaštićena od neovlaštenog pristupa ili izmjene. U sustav su stoga ugrađeni mehanizmi za provjeru prava pristupa do pojedinih usluga, zaštite tajnosti i vjerodostojnosti prenošenih podataka te zaštitu privatnosti sudionika u komunikaciji. Budući da je programskim modelom zasnovanim na uslugama omogućena hijerarhijska izgradnja složenih usluga povezivanjem pojedinačnih usluga, potreban je i prilagođeni mehanizam provjere prava pristupa do složenih usluga. Složenu uslugu njezini korisnicima ili druge usluge doživljavaju kao elementarnu uslugu. Prividna elementarnost složene usluge omogućuje jednostavnu izgradnju hijerarhijskih sustava zasnovanih na uslugama. U takvim je sustavima iznimno teško nadzirati prava pristupa do svake pojedinačne usluge zbog velikog broja pristupnih ograničenja koja je potrebno definirati i održavati. Za nadzor nad pravima pristupa u hijerarhijskim sustavima zasnovanim na uslugama osmišljen je hijerarhijski sustav provjere prava pristupa. Provjera pristupa obavlja se samo do usluge koju pozivatelj vidi kao elementarnu. U slučaju prividno elementarne usluge, pristup do pojedinačnih usluga od koje se ona sastoji definira se pristupnim pravima na nižoj razini hijerarhije, nevidljivoj vanjskom pozivatelju. Hijerarhijskim sustavom provjere prava pristupa očuvana je privatnost pozivajuće usluge ili korisnika i pozvane usluge jer su podaci o njihovim identitetima dostupni samo na najvišoj razini hijerarhije. Iscrpni opis funkcionalnosti, arhitekture i programskog ostvarenja podsustava za siguran pristup uslugama te hijerarhijskog sustava provjere prava pristupa do složenih usluga može se pronaći u [99, 100].

6 Prividna mreža računalnih sustava zasnovanih na uslugama

Programske usluge su osnovni gradivni elementi raspodijeljenih programskih sustava izgrađenih primjenom programskog modela zasnovanog na uslugama. Uobičajeni način raspodjele programskih funkcionalnosti u obliku programskih usluga je postupak raspodjele u komunikacijskom prostoru mreže Internet. Taj se način naziva raspodjelom u fizičkom komunikacijskom prostoru. Pristup uslugama u fizičkom komunikacijskom prostoru ostvaruje se poznavanjem njihovih jedinstvenih oznaka sredstva (engl. *Uniform Resource Locator – URL*) [101], čiji je sastavni dio i IP adresa [9] mrežnog čvora na kojem je usluga smještena. Druga mogućnost raspodjele programskih usluga jest oblikovanje prividne mreže i korištenje njezinog komunikacijskog prostora. Komunikacijski prostor prividne mreže naziva se logičkim komunikacijskim prostorom. Pristup uslugama u logičkom komunikacijskom prostoru prividne mreže ostvaruje se poznavanjem logičkih imena mrežnih čvorova te logičkih imena programskih usluga. Korištenje komunikacijskog prostora prividne mreže pojednostavljuje i olakšava razvoj i izvođenje novih te održavanje postojećih raspodijeljenih programskih sustava zasnovanih na uslugama.

6.1 Smjernice za oblikovanje i izgradnju prividne mreže

Osnovni zahtjev kojemu prividna mreža računalnih sustava zasnovanih na uslugama treba udovoljiti jest potpora oblikovanju, izgradnji, postavljanju u radnu okolinu te održavanju izgrađenih raspodijeljenih računalnih sustava zasnovanih na uslugama. Povezivanjem programskih usluga u raspodijeljene računalne sustave zasnovane na uslugama korištenjem onolikog broja mrežnih čvorova i sustava adresiranja usluga koji vrijede u fizičkoj mreži, izgrađeni sustavi postaju vezani i ovisni o organizaciji fizičke mreže u danoj okolini. Pri prijenosu izgrađenog sustava u drugu radnu okolinu potrebno je izvršiti prilagodbu sustava na uvjete koji vladaju u toj okolini. Prilagodba uvjetima nove radne okoline obuhvaća prilagodbu raspodijeljenih računalnih sustava novom sustavu adresiranja mrežnih čvorova te raspodjeli programskih usluga u skladu s brojem raspoloživih mrežnih čvorova. Prilagodba raspodijeljenih sustava uvjetima fizičkih radnih okolina složen je postupak kojemu, osim prilagodbe adresama i broju čvorova, prethodi planiranje načina raspodjele programskih usluga. Osim prilagodbe računalnog sustava uvjetima fizičke radne okoline, zadatak sustava prividne mreže jest osiguravanje mehanizama za izgradnju raspodijeljenih računalnih sustava koji su otporni na kvarove čvorova fizičke mreže. U

slučaju pojave kvara, sustav prividne mreže treba prilagoditi fizičku mrežu uklanjanjem pokvarenih čvorova, preseljenjem funkcionalnosti pokvarenih čvorova na ispravne čvorove te prikrivanjem poduzetih mjera od raspodijeljenog sustava u izvođenju. Potpora oblikovanju, izgradnji, postavljanju u radnu okolinu te održavanju raspodijeljenih sustava zasnovanih na uslugama očituje se kroz mogućnost prilagodbe prividne mreže s jedne strane prema raspodijeljenim sustavima, a s druge strane prema fizičkoj mreži koja je raspoloživa u danoj okolini.

Programske usluge u raspodijeljenom računalnom sustavu zasnovanom na uslugama istovremeno poprimaju ulogu korisnika i poslužitelja računalnih sredstava koje nude ili potražuju druge usluge. Programske usluge mogu biti pozvane od strane drugih usluga ili vanjskih korisničkih programa, dok one same za ostvarivanje svojih funkcionalnosti također mogu pozivati druge usluge. Komunikacijski prostor raspodijeljenih usluga, stoga, poprima obilježja mreže ravnopravnih sudionika. Prividna mreža raspodijeljenih računalnih sustava zasnovanih na uslugama treba udovoljavati uvjetima mreže ravnopravnih sudionika u smislu ravnopravnosti čvorova i dinamičnosti mreže. Svi čvorovi prividne mreže trebaju biti izjednačeni u funkcionalnostima. Svaki od čvorova treba biti u stanju slati i primati podatke, odnosno imati ulogu izvorišta i odredišta podataka. Za izgradnju složenih mehanizama sigurnog prijenosa podataka koji jamče privatnost sudionika mrežne komunikacije, prikrivajući njihov identitet, čvorovi prividne mreže trebaju biti u stanju prosljeđivati podatke u korist ostalih mrežnih čvorova. Budući da je oblikovanje i uspostavljanje prividne mreže raspodijeljene računalne okoline prepušteno graditeljima, održavateljima, odnosno krajnjim korisnicima raspodijeljenih programskeh sustava zasnovanih na uslugama, mehanizmi prividne mreže trebaju omogućiti dinamičko upravljanje prividnom mrežom za vrijeme njezina rada i rada čitavog sustava prividne raspodijeljene računalne okoline. Dinamičko upravljanje prividnom mrežom podrazumijeva mogućnost ulaska novih čvorova u mrežu, kao i mogućnost napuštanja mreže od strane čvorova koji već jesu članovi mreže.

Komunikacija među programskim uslugama raspodijeljenima u komunikacijskom prostoru prividne mreže treba biti neovisna i odvojena od komunikacije fizičkim mrežnim vezama. Nezavisnost prividne mreže o fizičkoj mreži postiže se odvajanjem komunikacijskih protokola logičke razine od onih koji se koriste za fizički prijenos podataka, odvajanjem sustava naslovljavanja usluga u prividnoj mreži od sustava adresiranja mrežnih čvorova i usluga u fizičkoj mreži te uspostavom logičkog ustroja prividne mreže (engl. *virtual network topology*) i sustava usmjeravanja poruka (engl. *routing*), također neovisnog o ustroju fizičke mreže i prijenosu podataka fizičkim komunikacijskim vezama. Komunikacijski protokoli za ostvarivanje komunikacije među programskim uslugama na logičkoj razini trebaju

ispunjavati svojstva nezavisnosti i sveobuhvatnosti. Svojstva nezavisnosti i sveobuhvatnosti nalažu da komunikacijski protokol logičke razine može iskorištavati prijenosni protokol fizičke razine za fizički prijenos poruka između dvaju fizičkih mrežnih čvorova, ali su informacije potrebne za pronalaženje komunikacijskog puta, uspostavu i odvijanje komunikacije sadržane u protokolu logičke razine. Poruke sastavljene prema nezavisnom i sveobuhvatnom komunikacijskom protokolu logičke razine uključuju podatke o naslovljavanju usluga i podatke potrebne za usmjeravanje poruka logičkim komunikacijskim vezama. Podaci o naslovljavanju usluga koriste se za osvarivanje pristupa, odnosno pozivanja programskih usluga. Podaci za usmjeravanje poruka primjenjuju se za uspostavu logičkog ustroja prividne mreže kojim se definira put poruke od izvorišne do odredišne programske usluge.

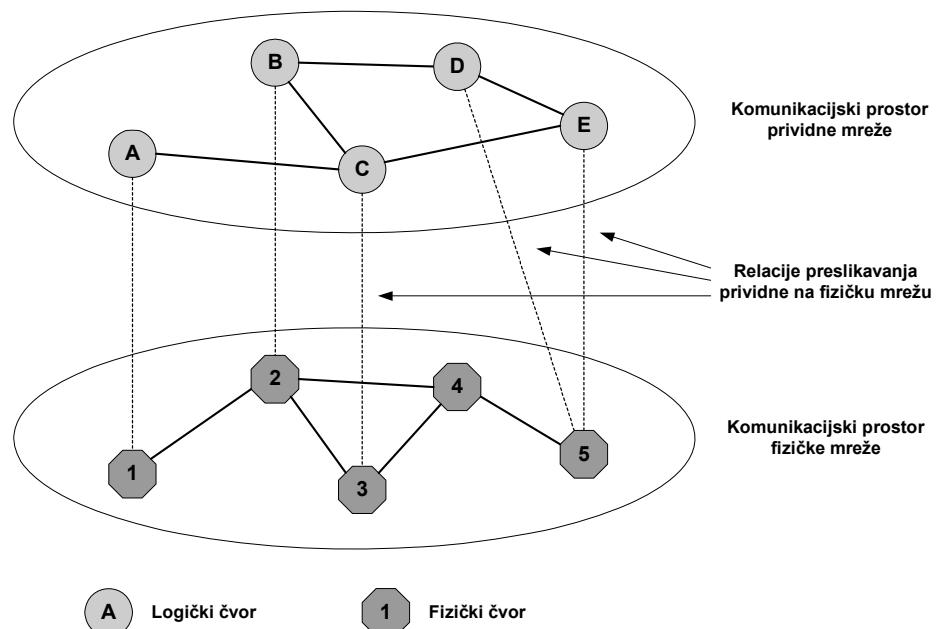
Budući da je prividna mreža rasprostranjena u otvorenom komunikacijskom prostoru mreže Internet, komunikacijskim protokolom logičke razine potrebno je osigurati mogućnost uspostavljanja sigurne komunikacije među udaljenim programskim uslugama. Sigurnosni mehanizmi ugrađeni u komunikacijski protokol logičke razine trebaju osigurati tajnost podataka sadržanih u prenošenim porukama, njihovu vjerodostojnost, izvornost njihova pošiljatelja, neporecivost djelovanja prouzročenog slanjem poruke te privatnost pošiljatelja i primatelja poruke. Tajnost podataka jamči se provedbom kriptografskih postupaka nad podacima u izvornom obliku. Vjerodostojnost podataka, izvornost pošiljatelja poruke te neporecivost djelovanja prouzročenog slanjem poruke očuvana je digitalnim potpisivanjem podataka. Privatnost sudionika u komunikaciji postiže se spregom kriptografskih postupaka i postupaka digitalnog potpisivanja, potpomognutima mehanizmima usmjeravanja poruka. Logičkim ustrojem prividne mreže i mogućnošću usmjeravanja poruka na logičkoj razini u sprezi s mehanizmima kriptiranja i digitalnog potpisivanja podataka moguće je zametnuti tragove izvorišta i odredišta poruke. Svi elementi sigurnog prijenosa podataka uglavljeni su u poruke komunikacijskog protokola prividne mreže, čime se potvrđuju njegova svojstva nezavisnosti i sveobuhvatnosti.

6.2 Mreža logičkih čvorova

Prividna mreža računalnih sustava zasnovanih na uslugama koja se koristi za izgradnju prividne raspodijeljene računalne okoline oblikovana je kao mreža logičkih čvorova. Logički čvorovi su nezavisni programski elementi koji imaju sposobnosti primanja, slanja i prosljeđivanja podataka među programskim uslugama. Osim triju spomenutih funkcionalnosti, logički čvorovi jamče i osnovne elementi sigurnog prijenosa podataka, kao

što su tajnost, privatnost, vjerodostojnost, izvornost i neporecivost komunikacije među programskim uslugama.

Logički čvorovi prividne mreže raspodijeljeni su po čvorovima fizičke mreže. Relacija preslikavanja logičkih čvorova na fizičke čvorove ima svojstvo preslikavanja više na jedan (engl. *many to one*). Jedan logički čvor smješten je na jednom fizičkom čvoru. S druge strane, na jedan fizički čvor moguće je smjestiti proizvoljno mnogo logičkih čvorova. Slikom 6.1 prikazana je prividna mreža s pet logičkih čvorova izgrađena nad fizičkom mrežom sastavljenom od pet fizičkih čvorova. Relacijom preslikavanja logičke na fizičku mrežu definirano je preslikavanje logičkih čvorova A, B i C na različite fizičke čvorove, dok su logički čvorovi D i E smješteni na istom fizičkom čvoru. U tako organiziranoj prividnoj mreži, jedan čvor fizičke mreže ostao je slobodan jer mu nije pridružen nijedan čvor prividne mreže.



Slika 6.1 Relacija preslikavanja prividne mreže na fizičku mrežu

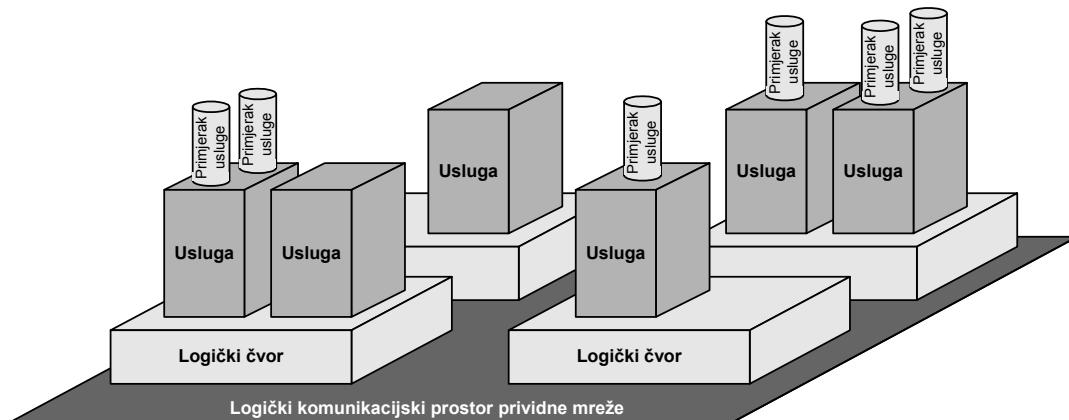
Logički čvorovi prividne mreže međusobno su povezani logičkim komunikacijskim vezama. Uspostavom logičkih komunikacijskih veza ostvaruje se komunikacija raspodijeljenih programskih usluga. Logičke veze među logičkim čvorovima prividne mreže programske su definirane apstraktne tvorevine koje oblikuju logički ustroj prividne mreže. O ustroju prividne mreže ovisi način usmjeravanja poruka među logičkim čvorovima. Dva logička čvora koja su međusobno povezana izravnim logičkim komunikacijskim vezama mogu na logičkoj razini ostvariti izravnu komunikaciju. Ako logički čvorovi nisu povezani

izravnim logičkim komunikacijskim vezama, njihova se komunikacija odvija posredstvom ostalih logičkih čvorova koji se nalaze na logičkom putu između tih dvaju čvorova. Čvorovi posrednici u tom slučaju poprimaju ulogu čvorova usmjernika poruka. Logički ustroj prividne mreže nezavisan je od stvarnog ustroja fizičke mreže, čime je i način komunikacije na logičkoj i fizičkoj razini bitno različit. Primjer na slici 6.1. pokazuje da su logički čvorovi A i C povezani izravnom logičkom komunikacijskom vezom, što znači da je na logičkoj razini među njima moguće uspostaviti izravnu komunikaciju. Fizički čvorovi 1 i 3, međutim, ne mogu izravno komunicirati jer u fizičkoj mreži ne postoji izravna veza koja bi povezala ta dva čvora. Fizički čvorovi 1 i 3 povezani su putem fizičkog čvora 2, koji pri komunikaciji čvorova 1 i 3 služi kao usmjernik poruka. Prema tome, logička komunikacija prividno izravnom vezom između logičkih čvorova A i C u stvarnosti se odvija uspostavom komunikacijskog puta između fizičkih čvorova 1 i 3 putem dvije fizičke komunikacijske veze, posredstvom fizičkog čvora 2. Logički čvor B koji je smješten na fizičkom čvoru 2, međutim, nije svjestan te komunikacije zbog nezavisnosti prividne mreže o fizičkoj mreži i odvojenosti komunikacijskih protokola na logičkoj i fizičkoj razini komunikacije.

6.3 Naslovljavanje usluga u komunikacijskom prostoru prividne mreže

Pristup uslugama u logičkom komunikacijskom prostoru prividne mreže ostvaruje se trorazinskim sustavom naslovljavanja. Sustav naslovljavanja usluga u komunikacijskom prostoru prividne mreže prikazan je slikom 6.2. Prvu razinu naslovljavanja čini naslovljavanje logičkog čvora u logičkom komunikacijskom prostoru prividne mreže na kojem je smještena usluga. Naslovljavanje čvorova prividne mreže raspodijeljene računalne okoline vrši se njihovim logičkim imenima. Sustav naslovljavanja logičkih čvorova ispunjava svojstva jedinstvenosti, jednostavnosti, privatnosti i nezavisnosti logičkih imena. Svakom logičkom čvoru pridruženo je jedinstveno logičko ime. Doseg valjanosti logičkog imena rasprostire se unutar logičkog komunikacijskog prostora prividne mreže. Imena logičkih čvorova oblikuju se kao prozvoljni nizovi tekstualnih znakova, čime oni postaju jednostavnii za uporabu i lako pamtljivi od strane čovjeka, odnosno graditelja raspodijeljenih programskih sustava. Imena logičkih čvorova neovisna su o adresama fizičkih čvorova na kojima se nalaze te o imenima i namjeni programskih usluga koje su postavljene po tim logičkim čvorovima. Nezavisnost imena logičkih čvorova o adresama fizičkih mrežnih čvorova omogućava proizvoljni način pridruživanja logičkih čvorova fizičkim čvorovima, odnosno oblikovanje proizvoljne relacije preslikavanja prividne na fizičku mrežu. Nepostojanje ograničenja pri oblikovanju relacije preslikavanja omogućava učinkovitu

provedbu postupaka preseljavanja logičkih čvorova među različitim fizičkim čvorovima. Na taj način prividna mreža pruža potporu za prenosivost izgrađenih programskih sustava zasnovanih na uslugama i njihovu otpornost na kvarove čvorova fizičke mreže.



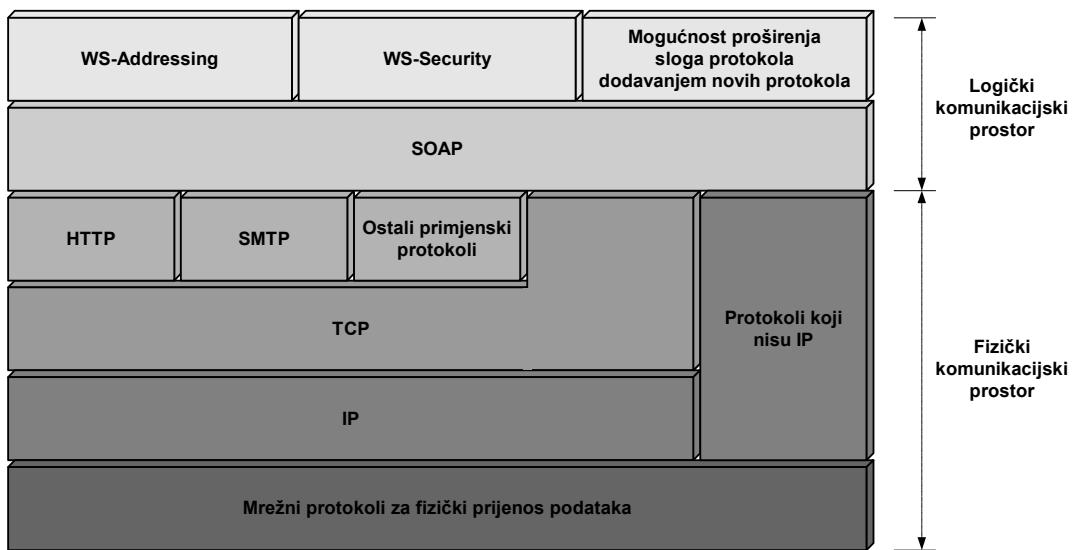
Slika 6.2 Trorazinska organizacija sustava naslovljavanja usluga u logičkom komunikacijskom prostoru prividne mreže

Drugu razinu naslovljavanja čini naslovljavanje usluge u komunikacijskom prostoru logičkog čvora. Naslovljavanje usluge obavlja se uporabom logičkog imena usluge. Za usluge koje se nalaze na istom logičkom čvoru vrijedi svojstvo jedinstvenosti logičkog imena usluge. U logičkom komunikacijskom prostoru prividne mreže može postojati veći broj usluga s istim logičkim imenima sve dok se one nalaze na različitim logičkim čvorovima.

S gledišta načina naslovljavanja, u komunikacijskom prostoru prividne mreže postoje dvije vrste programskih usluga. Usluge bez očuvanja stanja su programske usluge koje svaki zahtjev obrađuju zasebno i neovisno o prethodno obrađenim zahtjevima. Usluge s očuvanjem stanja su programske usluge u kojima obrada zahtjeva ovisi o rezultatima prethodno obrađenih zahtjeva. Usluge s očuvanjem stanja koriste primjerke usluge (engl. *service instances*) za spremanje stanja. Pristup uslugama bez očuvanja stanja potpuno je određen s dvije razine naslovljavanja. Za pristup uslugama s očuvanjem stanja koristi se treća razina naslovljavanja. Pristup do usluga s očuvanjem stanja obavlja se pristupanjem određenom primjerku usluge. Pristup do primjerka usluge ostvaruje se naslovljavanjem željenog primjerka. Naslovljavanje primjeraka usluga obavlja se imenima primjeraka. Odnosi između imena primjerka i imena usluge istovjetni su odnosima između imena usluge i imena logičkog čvora. Sve dok primjeri usluga pripadaju različitim uslugama, za njihovo je naslovljavanje moguće koristiti ista imena.

6.4 Logički komunikacijski protokol prividne mreže

Pristup uslugama raspodijeljenima u komunikacijskom prostoru prividne mreže i njihova međusobna komunikacija odvojena je i neovisna o komunikacijskim protokolima i postupcima adresiranja koji se koriste u fizičkoj mreži nad kojom je prividna mreža uspostavljena. Neovisnost prividne mreže raspodijeljene računalne okoline o fizičkoj mreži postignuta je sloganom protokola koji ocrtava jasno razdvajanje funkcionalnosti i protokola koji pripadaju logičkom, odnosno fizičkom komunikacijskom prostoru. Slikom 6.3 prikazan je slogan protokola prividne mreže te podjela protokola na fizički i logički komunikacijski prostor, promatrano sa stajališta korisnika prividne raspodijeljene računalne okoline.



Slika 6.3 Slog protokola prividne mreže raspodijeljene računalne okoline

Na najnižoj razini sloga protokola ostvareni su protokoli za ostvarivanje fizičkog prijenosa podataka prijenosnim medijima. Medij za prijenos podataka može biti električni vodič pri ožičenim mrežama, odnosno zrak pri mrežama s bežičnim tehnologijama prijenosa. Iznad protokola za fizički prijenos podataka nalazi se skupina mrežnih protokola. Mrežni protokoli osiguravaju usmjeravanje podataka kroz mrežu pronalaskom najpovoljnijih komunikacijskih putova između izvorišnog i odredišnog mrežnog čvora, sprječavaju nastajanje zagušenja u mreži uslijed prevelikog opterećenja mrežnih čvorova i prijenosnih veza i slično. Globalna računalna mreža Internet kao okosnica izgradnje prividne raspodijeljene računalne okoline koristi *IP* (engl. *Internet Protocol*) protokol [9] kao osnovni mrežni protokol. Ostale mreže koriste sebi svojstvene vrste mrežnih protokola. Na sloj mrežnih protokola, odnosno na *IP* protokol u slučaju mreže Internet, naslanja se sloj

prijenosnih protokola. Prijenosni protokoli osiguravaju pouzdanu isporuku podataka odredišnim čvorovima, sprječavajući njihov gubitak i višestruke isporuke. Pouzdani prijenos podataka u mreži Internet ostvaruje se primjenom *TCP* (engl. *Transport Control Protocol*) protokola [9]. *TCP* protokol moguće je izravno iskoristiti za prijenos podataka primjenskih programa. Njegova je općenitost, međutim, vrlo često nedovoljno izražajna te su za prijenos podataka primjenske razine razvijeni brojni primjenski protokoli. Najširu primjenu u sustavima zasnovanim na uslugama nalazi *HTTP* (engl. *HyperText Transfer Protocol*) protokol [102] i *SMTP* (engl. *Simple Mail Transport Protocol*) protokol [103]. Primjenskim protokolima mreže Internet završava fizički komunikacijski prostor prividne mreže računalnih sustava zasnovanih na uslugama.

Okosnicu logičkog komunikacijskog prostora prividne raspodijeljene računalne okoline čini *SOAP* protokol [20]. *SOAP* protokol je standardom dogovoren način prijenosa podataka i pristupa programskim uslugama u sustavima zasnovanim na uslugama. Zasnovan je na *XML* jeziku [18, 19], što ga čini neovisnim o prijenosnom protokolu iz fizičkog komunikacijskog prostora. U logičkom komunikacijskom prostoru prividne mreže, *SOAP* protokol ima ulogu logičkog prijenosnog protokola. Stvarni prijenosni protokol u odnosu na *SOAP* protokol je protokol iz fizičkog komunikacijskog prostora koji se nalazi neposredno ispod *SOAP* protokola. Trenutna izvedba prividne mreže koristi *HTTP* protokol kao fizički prijenosni protokol za prijenos logičkog *SOAP* protokola.

Jedinice prijenosa podataka koje se prenose *SOAP* protokolom nazivaju se *SOAP* porukama. Korištenje *XML* jezika za oblikovanje *SOAP* poruka omogućuje jednostavno proširivanje osnovnog oblika protokola. Proširenjem osnovnog oblika *SOAP* poruka nastaju protokoli logičke primjenske razine. Prividna mreža računalnih sustava zasnovanih na uslugama koristi dva proširenja osnovnog *SOAP* protokola. *WS-Addressing* specifikacija [105] standardom je usvojen način za definiranje *XML* struktura kojima se *SOAP* poruke proširuju podacima o naslovljavanju izvorišnih i odredišnih programskih usluga. Poruke oblikovane prema *WS-Addressing* specifikaciji sadrže sve informacije potrebne za prijenos poruke od izvorišne do odredišne programske usluge, bez potrebe za korištenjem podataka iz nižih protokola fizičke razine. Primjenom *WS-Addressing* specifikacije je, stoga, omogućeno stvaranje logičkog komunikacijskog prostora prividne mreže koji je u potpunosti neovisan o fizičkoj mreži i protokolima fizičkog komunikacijskog prostora. *WS-Security* specifikacija [106] drugo je proširenje osnovnog *SOAP* protokola koje koristi prividna mreža računalnih sustava zasnovanih na uslugama. *WS-Security* specifikacija propisuje način oblikovanja *XML* struktura za zaštitu podataka sadržanih u *SOAP* porukama te njihovu ugradnju u *SOAP* poruke. Primjenom *WS-Security* specifikacije omogućeno je očuvanje tajnosti (engl.

confidentiality) i vjerodostojnosti (engl. *integrity*) podataka sadržanih u *SOAP* porukama, utvrđivanje izvornosti (engl. *authenticity*) pošiljatelja *SOAP* poruka te jamčenje neporecivosti (engl. *non-repudiation*) djelovanja prouzročenog slanjem *SOAP* poruka. Spregom *WS-Addressing* i *WS-Security* specifikacija postiže se prikrivanje identiteta (engl. *privacy*) sudionika koji za komunikaciju koriste *SOAP* protokol. Osim *WS-Addressing* i *WS-Security* specifikacija, postoje i druge specifikacije logičke primjenske razine koji se u sustavima zasnovanim na uslugama koriste u različite svrhe. Za potrebe mogućih novih primjena, proširivost osnovnog oblika *SOAP* protokola omogućuje njihovo jednostavno objedinjavanje s postojećim proširenjima.

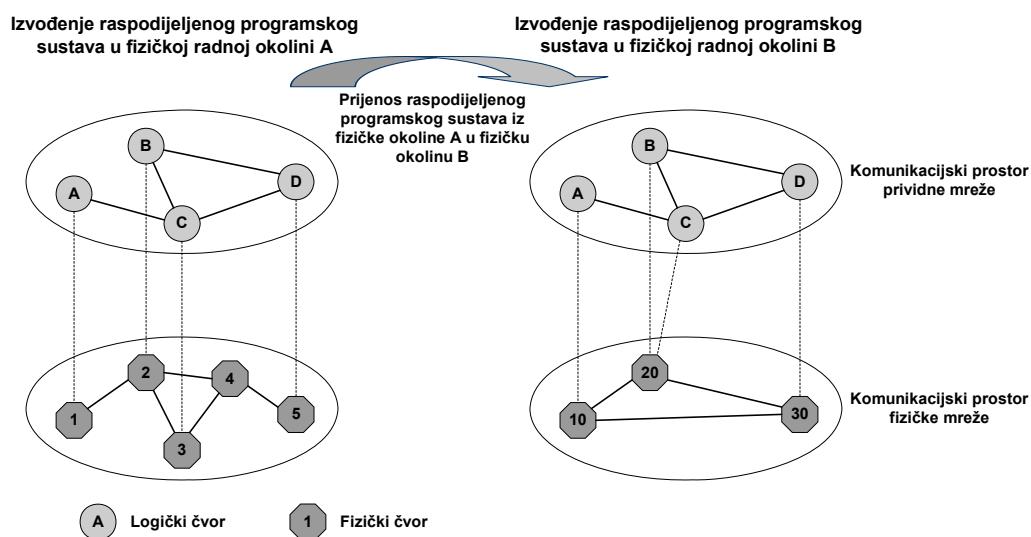
6.5 Prenosivost raspodijeljenih programskih sustava

Nezavisnošću o fizičkoj mreži nad kojom je izgrađena, prividna mreža postaje osnovica za potporu prenosivosti raspodijeljenih programskih sustava zasnovanih na uslugama te oblikovanje i izgradnju mehanizama otpornosti programskih usluga i raspodijeljenih programskih sustava na kvarove mrežnih čvorova. Raspodijelom programskih usluga te izgrađivanjem raspodijeljenih programskih sustava u komunikacijskom prostoru prividne mreže u potpunosti neovisne o fizičkoj mreži, izgrađeni raspodijeljeni programski sustavi također postaju neovisni o fizičkoj mreži. Neovisnost raspodijeljenih programskih sustava o fizičkoj mreži očituje se u njihovoj nezavisnosti o adresama fizičkih čvorova na kojima su raspodijeljene programske usluge koje ih sačinjavaju te u broju fizičkih čvorova na kojima su te usluge raspodijeljene.

Raspodijeljeni programski sustavi zasnovani na uslugama za pozivanje primjenskih usluga ili usluga za suradnju i natjecanje koriste imena ili označe sredstva programskih usluga te adrese mrežnog čvora na kojima su pozvane usluge smještene. Ako se adresiranje programskih usluga provodi u komunikacijskom prostoru fizičke mreže, koristeći fizičke adrese čvorova fizičke mreže, raspodijeljeni programi su čvrsto vezani uz trenutno stanje fizičke mreže. Čvrsta povezanost raspodijeljenih programa sa stanjem fizičke mreže očituje se u ovisnosti poziva programskih usluga o adresnom prostoru fizičke mreže i raspodjeli tih usluga po čvorovima fizičke mreže koja vrijedi u trenutku stvaranja raspodijeljenog programa. Ako se za vrijeme životnog vijeka raspodijeljenog programskog sustava zasnovanog na uslugama dogode promjene u načinu adresiranja čvorova fizičke mreže, njihovom broju i raspodijeli programskih usluga, izmjene je potrebno provesti u svim raspodijeljenim programima. Isti se problem javlja i pri pokušaju prijenosa raspodijeljenog programskog sustava iz jedne fizičke radne okoline u drugu. Pri promjeni radne okoline

raspodijeljenog programskog sustava, problem nepodudarnosti broja i sustava adresiranja fizičkih čvorova je neizbjegjan.

Problem prenosivosti raspodijeljenih programskih sustava zasnovanih na uslugama rješava se njihovim oblikovanjem i izgradnjom u logičkom komunikacijskom prostoru prividne mreže. Postupku oblikovanja i izgradnje raspodijeljenog programskog sustava zasnovanog na uslugama prethodi postupak oblikovanja prividne mreže. Analizom svojstava učinkovitosti i sigurnosti koje raspodijeljeni programski sustav treba zadovoljiti, utvrđuje se broj logičkih čvorova potreban za raspodijeljivanje programskih usluga koje ostvaruju njegove funkcionalnosti. Programsko ostvarenje logike za usklađivanje rada primjenskih usluga u obliku raspodijeljenih programa tada se provodi u komunikacijskom prostoru prividne mreže. Korištenje komunikacijskog prostora prividne mreže omogućuje zamjenu adresa čvorova fizičke mreže logičkim imena čvorova prividne mreže i njihovo korištenje pri pozivanju programskih usluga. Budući da su logička imena čvorova prividne mreže nezavisna od adresa fizičkih čvorova, raspodijeljeni programi postaju nezavisni o adresnom sustavu fizičke mreže.



Slika 6.4 Prenosivost raspodijeljenih programskih sustava izgrađenih u logičkom komunikacijskom prostoru prividne mreže

Postupak prenošenja raspodijeljenog programskog sustava zasnovanog na uslugama iz jedne radne okoline u drugu prikazan je slikom 6.4. Za raspodijeljeni programski sustav izgrađen u logičkom komunikacijskom prostoru prividne mreže pri promjeni fizičke radne okoline ne postoje razlike u sustavu adresiranja, broju čvorova i ustrojstvu fizičke mreže. Razlike u fizičkim mrežama dviju radnih okolina prikriva sustav prividne mreže. Prilagodbom relacije preslikavanja prividne na fizičku mrežu, dva različita fizička

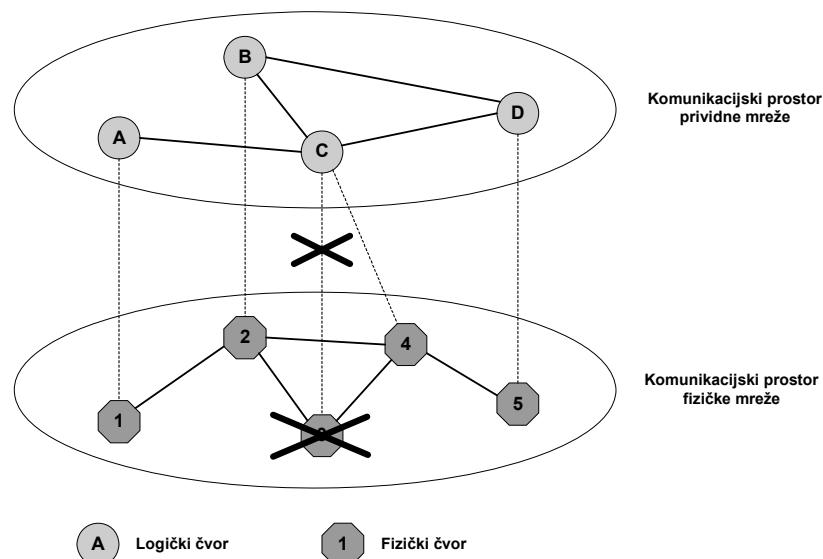
komunikacijska prostora pretvaraju se u potpuno jednake logičke komunikacijske prostore. U primjeru na slici 6.4 raspodijeljeni programski sustav izgrađen je u logičkom komunikacijskom prostoru prividne mreže koja se sastoji od četiri logička čvora. U radnoj okolini A postoji fizička mreža sastavljena od pet fizičkih čvorova. Relacijom preslikavanja koja vrijedi u radnoj okolini A definirano je preslikavanje logičkih čvorova A, B, C i D na fizičke čvorove čije se adrese redom 1, 2, 3 i 5. Pri prijenosu raspodijeljenog programskog sustava u radnu okolinu B na raspolaganju je fizička mreža koju čine tri fizička čvora. Osim promjena u adresama fizičkih čvorova, prijenos raspodijeljenog programskog sustava u radnu okolinu B suočava se s problemom manjka jednog fizičkog čvora. Logički komunikacijski prostori dviju radnih okolina izjednačuju se prilagodbom relacije preslikavanja prividne na fizičku mrežu. Relacijom preslikavanja koja vrijedi u novoj radnoj okolini definirano je preslikavanje logičkih čvorova A, B, C i D na čvorove fizičke mreže čije su adrese redom 10, 20, 20 i 30. Problem manjka jednog fizičkog čvora u novoj je radnoj okolini riješen smještanjem dvaju logičkih čvorova na isti fizički čvor.

6.6 Otpornost raspodijeljenih programskih sustava na kvarove mrežnih čvorova

Sustav prividne mreže u suradnji sa sustavom za postavljanje usluga prividne raspodijeljene računalne okoline omogućuje oblikovanje i izgradnju mehanizama otpornosti programskih usluga i raspodijeljenih programskih sustava zasnovanih na uslugama na kvarove čvorova fizičke mreže. Relacija preslikavanja prividne na fizičku mrežu koja dozvoljava pridruživanje većeg broja logičkih čvorova jednom fizičkom čvoru osnovni je mehanizam kojim je poduprta izgradnja takvih mehanizama. Sve dok su programske usluge raspodijeljene u logičkom komunikacijskom prostoru prividne mreže, prilagođavanjem relacije preslikavanja svaki je dio logičkog komunikacijskog prostora moguće preslikati na bilo koji čvor fizičke mreže. To se svojstvo iskorištava tijekom oblikovanja i izgradnje mehanizama otpornosti programskih usluga i raspodijeljenih programskih sustava zasnovanih na uslugama na kvarove čvorova fizičke mreže.

Načelni prikaz potpore sustava prividne mreže izgradnji mehanizama otpornosti programskih usluga i raspodijeljenih programskih sustava zasnovanih na uslugama na kvarove čvorova fizičke mreže prikazan je slikom 6.5. Raspodijeljeni programski sustav izgrađen je u logičkom komunikacijskom prostoru prividne mreže koja se sastoji od četiri logička čvora čija su logička imena redom A, B, C i D. Na početku rada sustava, nad prividnom mrežom je definirana relacija preslikavanja na fizičku mrežu kojom se logički

čvorovi A, B, C i D preslikavaju na fizičke čvorove čije su adrese redom 1, 2, 3 i 5. Za vrijeme rada sustava dolazi do kvara na fizičkom čvoru s adresom 3. Da bi raspodijeljeni programski sustav ostao neosjetljiv na kvar koji se dogodio, sve programske usluge smještene na pokvarenom fizičkom čvoru koje on koristi treba postaviti na neki od ispravnih fizičkih čvorova. Nakon toga je potrebno sve pozive prema tim uslugama preusmjeriti na novi fizički čvor. Korištenjem prividne mreže i sustava za postavljanje usluga, ti su zahvati znatno pojednostavljeni. U trenutku otkrivanja kvara na fizičkom čvoru 3, sustav za postavljanje usluga pretraživanjem imenika usluga pronađe sve usluge koje su bile postavljene na tom čvoru. Zapise tih usluga briše iz imenika usluga te sve usluge ponovno postavlja na novi fizički čvor, primjerice onaj s adresom 4. Sustav prividne mreže nakon toga promjenom relacije preslikavanja prividne na fizičku mrežu logičko ime čvora C, umjesto pokvarenom fizičkom čvoru na adresi 3, pridružuje ispravnom fizičkom čvoru koji se nalazi na adresi 4.



Slika 6.5 Upravljanje otpornošću na kvarove čvorova fizičke mreže programskim izmjenama relacije preslikavanja prividne na fizičku mrežu

Spregom sustava za postavljanje usluga i sustava prividne mreže, kvarove čvorova fizičke mreže moguće je otklanjati veći broj puta. U konačnici je moguće sve logičke čvorove prividne mreže svesti na samo jedan fizički čvor. Zanemare li se svojstva učinkovitosti raspodijeljenog programskog sustava koja mogu biti narušena zbog preopterećenosti fizičkog čvora, cjelokupni raspodijeljeni računalni sustav ispunjava svoju funkciju sve dok je barem jedan čvor fizičke mreže u radnom stanju.

6.7 Potpora prostornoj lokalnosti usluga

Prilikom oblikovanja prividne mreže za izgradnju raspodijeljenog sustava zasnovanog na uslugama postavlja se pitanje brojnosti logičkih čvorova i pitanje raspodjele programskih usluga po logičkim čvorovima. Najvažnija mjerila za oblikovanje sustava prividne mreže su učinkovitost i sigurnost izgrađenog raspodijeljenog sustava koji se u toj mreži izvodi te jednostavnost provedbe postupaka prenosivosti izgrađenih sustava i njihove otpornosti na kvarove čvorova fizičke mreže. Budući da je zrnatost raspodijeljenog računalnog sustava zasnovanog na uslugama uspostavljena na razini programskih usluga, stvaranje zasebnog logičkog čvora za svaku uslugu od koje se sastoji raspodijeljeni sustav omogućuje najlakši i najjednostavniji način provedbe postupaka prenosivosti i otpornosti na kvarove. S druge strane, općenito pravilo koje vrijedi za oblikovanje učinkovitih raspodijeljenih sustava jest raspodijeljivanje programskih funkcionalnosti na način da mrežna komunikacija među udaljenim dijelovima sustava nema značajnog utjecaja na ukupno vrijeme izvođenja. Što se tiče sigurnosti raspodijeljenih sustava, njihova je sigurnost najčešće ugrožena pri prijenosu osjetljivih podataka računalnom mrežom. Pravilo za izbjegavanje visokih sigurnosnih rizika pri izgradnji raspodijeljenih računalnih sustava nalaže izbjegavanje prijenosa osjetljivih podataka računalnom mrežom u što je moguće većoj mjeri.

Zahtjev za jednostavnošću provedbe postupaka prenosivosti i otpornosti na kvarove u suprotnosti je sa svojstvima učinkovitosti i sigurnosti raspodijeljenih računalnih sustava. Pomirenje svih triju zahtjeva postiže se odgovarajućim kompromisima i tu do izražaja najviše dolazi domisljatost graditelja raspodijeljenih računalnih sustava. Pri stvaranju kompromisa između triju oprečnih zahtjeva, graditelju raspodijeljenih računalnih sustava pomaže potpora sustava prividne mreže izgradnji učinkovitih i sigurnih programskih rješenja. Oba važna svojstva raspodijeljenih programskih sustava zasnovanih na uslugama, svojstvo učinkovitosti i svojstvo sigurnosti, moguće je ispuniti vodeći računa o prostornoj lokalnosti usluga. Usluge koje zahtijevaju učestalu međusobnu komunikaciju uputno je smjestiti na isti mrežni čvor. Iako su funkcionalnosti i dalje logički raspodijeljene, poziv usluge na lokalnom mrežnom čvoru znatno je učinkovitiji od poziva usluge na udaljenom čvoru. Slično razmatranje vrijedi i za zadovoljavanje svojstva sigurnosti. Usluge kojima je svojstvena učestala izmjena podataka podložnih visokim sigurnosnim rizicima također je preporučljivo smjestiti na isti mrežni čvor.

Načelo organizacije prividne mreže u obliku skupa logičkih čvorova pruža potporu za izgradnju učinkovitih i sigurnih raspodijeljenih programskih sustava zasnovanih na uslugama. Relacija preslikavanja prividne na logičku mrežu osigurava cjelovitost logičkog

čvora, što znači da se sve programske usluge koje su dostupne na jednom logičkom čvoru uvijek nalaze na istom fizičkom čvoru. Pravilo za oblikovanje prividne mreže za potporu izgradnji i izvođenju učinkovitih i sigurnih raspodijeljenih sustava zasnovanih na uslugama nalaže postavljanje programskih usluga među kojima se odvija učestala komunikacija, kao i onih koje razmjenjuju sigurnosno osjetljive podatke na iste logičke čvorove.

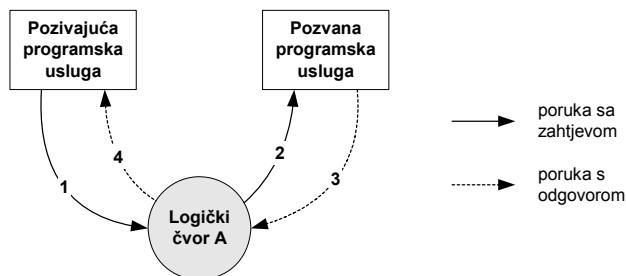
6.8 Usmjeravanje poruka

Trorazinski sustav naslovljavanja programskih usluga koji je u uporabi u komunikacijskom prostoru prividne mreže koristi se za usmjeravanje poruka i za ostvarivanje pristupa uslugama. Prva razina sustava naslovljavanja koristi se u postupku usmjeravanja poruka između čvorova prividne mreže, od izvorišnog do odredišnog logičkog čvora. Preostale dvije razine koriste se za pristup programskim uslugama na odredišnom čvoru prividne mreže.

Usmjeravanje sadržaja u logičkom komunikacijskom prostoru prividne mreže obavlja se na osnovi imena logičkih čvorova. Svaki logički čvor prividne mreže može istovremeno poprimiti tri različite uloge. Osim uobičajenih uloga pošiljatelja, odnosno primatelja poruke, logički čvor može poprimiti i ulogu usmjernika poruke. Nakon što zaprimi poruku, logički čvor ispituje logičko ime odredišnog logičkog čvora koje je sadržano u primljenoj poruci. Ako je logičko ime odredišnog logičkog čvora jednakom njegovom vlastitom logičkom imenu, logički čvor zaključuje da je poruku potrebno isporučiti nekoj od programskih usluga smještenih na njemu samome. Za isporuku poruke lokalnoj programskoj usluzi, logički čvor koristi drugu razinu naslovljavanja usluga. Druga razina naslovljavanja sadrži logičko ime programske usluge kojoj je poruka namijenjena. Iz primljene poruke logički čvor dohvaca logičko ime pozvane usluge te joj isporučuje poruku. Ako pozvana usluga ima obilježja usluge s očuvanjem stanja, koristi se i treća razina naslovljavanja, odnosno ime primjerka usluge. Isporuka poruke odgovarajućem primjerku usluge nije u nadležnosti logičkog čvora, već sama programska usluga vodi podatke o svojim aktivnim primjercima i odgovorna je za pravilnu isporuku poruke naslovljenom primjerku.

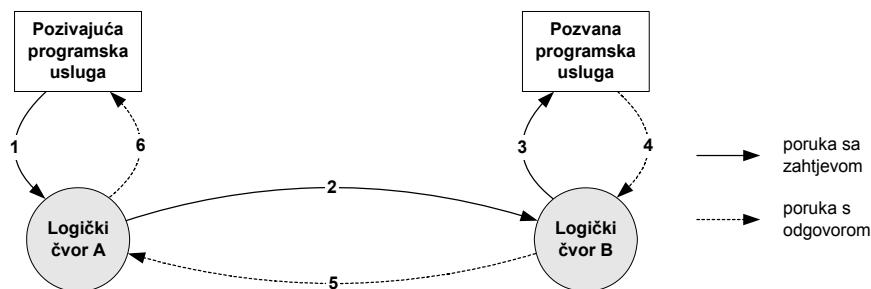
Komunikacija dviju programskih usluga uvijek se odvija posredstvom logičkih čvorova, bez obzira nalaze li se usluge na istom ili različitim logičkim čvorovima. Time se osigurava jedinstven način pristupanja programskim uslugama. Tijekom pozivanja programske usluge, pozivajuća usluga upućuje poruku s pozivom lokalnom logičkom čvoru na kojem je smještena, navodeći pritom logička imena čvora i usluge koju poziva. Slikom

6.6 prikazan je postupak usmjeravanja poruka za slučaj kada su obje programske usluge, pozivajuća i pozvana, smještene na istom logičkom čvoru.



Slika 6.6 Poziv programske usluge na lokalnom logičkom čvoru

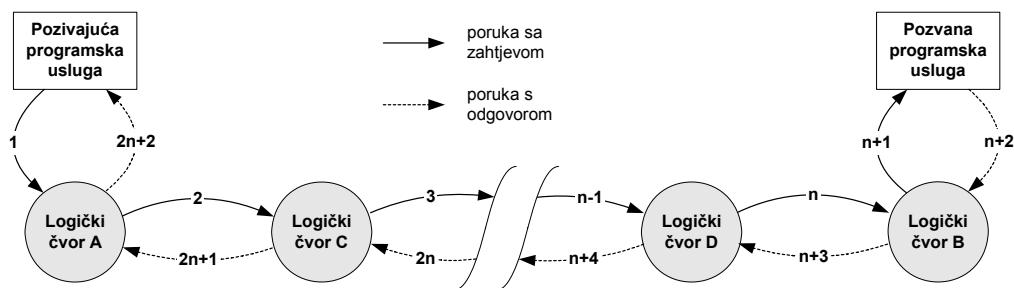
Iako je moguć slučaj jednosmjerne komunikacije, poziv programske usluge u većini se slučajeva sastoji od upućivanja zahtjeva za pozivom udaljene usluge i primanja odgovora s rezultatima njezina izvođenja. Pozivajuća programska usluga na logičkom čvoru A u koraku 1 poruku sa zahtjevom naslovljrenom za pozvanu uslugu koja se također nalazi na logičkom čvoru A šalje svom lokalnom logičkom čvoru. Logički čvor A na osnovi podudarnosti odredišnog i lokalnog logičkog imena zaključuje da se poziva lokalno smještena usluga, dohvata iz poruke njezino logičko ime te joj isporučuje poruku u koraku 2. Slanje odgovora pozivajućoj usluzi u koracima 3 i 4 slijedi isti postupak, ali u obrnutom smjeru.



Slika 6.7 Poziv programske usluge na udaljenom logičkom čvoru izravnom komunikacijom logičkih čvorova

Ako se ispitivanjem logičkog imena odredišnog logičkog čvora utvrdi da poruka nije namenjena programskoj usluzi koja se nalazi na lokalnom logičkom čvoru, onda je poruku potrebno proslijediti do odredišnog logičkog čvora. Prosljeđivanje poruke odredišnom logičkom čvoru moguće je izvesti na dva načina. Jednostavnijim načinom prosljeđivanja, poruka se proslijeđuje izravno odredišnom logičkom čvoru. Takav je slučaj prikazan slikom 6.7. Pozivajuća usluga smještena na logičkom čvoru A poziva uslugu smještenu na logičkom čvoru B. Logički čvor A tijekom ispitivanja logičkog imena odredišnog čvora utvrđuje da

primljena poruka nije namjenjena lokalnoj programskoj usluzi, već ju je potrebno prosljediti dalje. Na osnovi logičkog imena odredišnog čvora, čvor A prosljeđuje poruku čvoru B, koji je nakon toga isporučuje lokalnoj programskoj usluzi. Iako se oba logička čvora u stvarnosti mogu nalaziti na istom fizičkom čvoru, komunikacija dviju programskih usluga ima obilježja komunikacije fizički udaljenih čvorova.



Slika 6.8 Poziv programske usluge na udaljenom logičkom čvoru slanjem poruka putem nekoliko logičkih čvorova usmjernika

Složeniji način usmjeravanja poruka između pozivajuće i pozvane programske usluge jest neizravno slanje poruka između izvořišnog i odredišnog logičkog čvora posredstvom nekoliko logičkih čvorova usmjernika. Postupak komunikacije dviju programskih usluga prosljeđivanjem poruka putem nekoliko čvorova usmjernika prikazan je slikom 6.8. Razlika u odnosu na prethodni slučaj jest u tome što čvor koji primi poruku, nakon ispitivanja logičkog imena odredišnog čvora ne upućuje poruku izravno odredišnom čvoru, već logičko ime čvora kojemu upućuje poruku dohvata iz tablice usmjeravanja. Svaki logički čvor održava vlastitu tablicu usmjeravanja. U tablici usmjeravanja se za svaki odredišni čvor nalazi podatak o čvoru usmjerniku putem kojeg je potrebno nastaviti usmjeravanje poruke. U primjeru sa slike 6.8, logički čvor A u svojim tablicama usmjeravanja pronalazi podatak da je poruku za logički čvor B potrebno prosljediti logičkom čvoru C. Čvor C usmjeravanje poruke do istog odredišnog čvora obavlja putem čvora D. Konačno, čvor D poruku za čvor B usmjerava putem čvora B. Čvor B je odredišni logički čvor te primljenu poruku isporučuje pozvanoj programskoj usluzi. Broj čvorova usmjernika u komunikacijskom lancu između izvořišnog i odredišnog logičkog čvora može biti proizvoljno velik. Iako je ovaj postupak složeniji od prethodno opisanog postupka s izravnim upućivanjem poruke odredišnom logičkom čvoru, može se pojednostaviti. Oblikovanjem tablice usmjeravanja s pravilima usmjeravanja u kojima su logička imena odredišnih logičkih čvorova jednaka logičkim imenima čvorova usmjernika, ovaj se postupak svodi na prethodni.

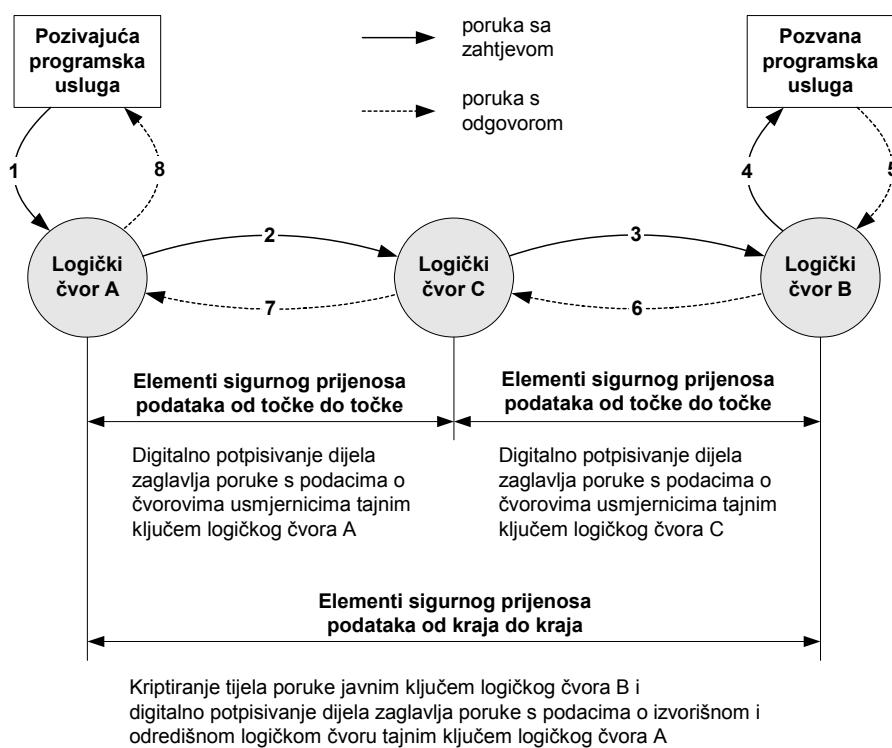
Mogućnost usmjeravanja poruka putem nekoliko čvorova usmjernika prije njegina pristizanja na odredište osnovni je mehanizam potpore privatnosti korisnika i pružatelja

programskih usluga. Kriptografskim operacijama nad imenima logičkih čvorova i programskih usluga, od vanjskih se promatrača skriva identitet pozivajuće i pozvane usluge te logičkih čvorova na kojima su smještene. Usmjeravanjem poruke putem nekoliko čvorova usmjernika, vanjski promatrač koji nije upoznat s ustrojstvom prividne mreže ne može dokučiti izvorište i krajnje odredište poruke. Presretanjem poruke između dvaju fizičkih čvorova, promatrač ne može utvrditi da li je poruka rezultat komunikacije programskih usluga na tim fizičkim čvorovima, ili je posrijedi komunikacija čvorova usmjernika koji su smješteni na promatranim fizičkim čvorovima. Dodatna mogućnost prikrivanja identiteta korisnika i pružatelja programskih usluga, kao i spriječavanje presretanja povjerljivih podataka jest mogućnost slanja zahtjeva i odgovora programskih usluga različitim komunikacijskim putovima. Ovim je postupkom, naime, omogućeno oblikovanje tablica usmjeravanja kojim se put poruke od izvorišnog do odredišnog logičkog čvora razlikuje od puta u obrnutom smjeru.

6.9 Sigurnost komunikacije

Elementi sigurnog prijenosa podataka u sustavu prividne mreže obuhvaćaju postupke očuvanja tajnosti (engl. *confidentiality*) i vjerodostojnosti (engl. *integrity*) prenošenih informacija, utvrđivanja izvornosti pošiljatelja poruke (engl. *authenticity*), nemogućnosti poricanja djelovanja prouzročenog slanjem poruke (engl. *non-repudiation*), ograničavanja prava pristupa do programskih usluga (engl. *access control*) te očuvanja privatnosti korisnika i pružatelja usluge (engl. *privacy*). Ispunjavanje navedenih elemenata sigurnog prijenosa podataka postiže se primjenom metoda kriptiranja i postupaka digitalnog potpisivanja nad podacima u izvornom obliku.

Oblikovanje sustava za siguran prijenos podataka između logičkih čvorova prividne mreže velikim je dijelom određeno načinom rada sustava za usmjeravanje poruka. Osim uobičajenog načina zaštite sigurnosti podataka između dviju krajnjih točaka koje sudjeluju u komunikaciji, mogućnost usmjeravanja poruka od izvorišta do odredišta putem većeg broja čvorova usmjernika zahtijeva i uvođenje elemenata sigurnog prijenosa podataka između pojedinih čvorova usmjernika. Elementi sigurnog prijenosa podataka između krajnjih točaka komunikacije nazivaju se elementima sigurnosti od kraja do kraja (engl. *end-to-end*), dok se elementi sigurnog prijenosa podataka između pojedinih usmjeravajućih točaka nazivaju elementima sigurnosti od točke do točke (engl. *point-to-point*). Slikom 6.9 prikazan je model sigurnog prijenosa podataka u sustavu prividne mreže s istaknutim dosezima djelovanja sigurnosnih elemenata od kraja do kraja i sigurnosnih elemenata od točke do točke.



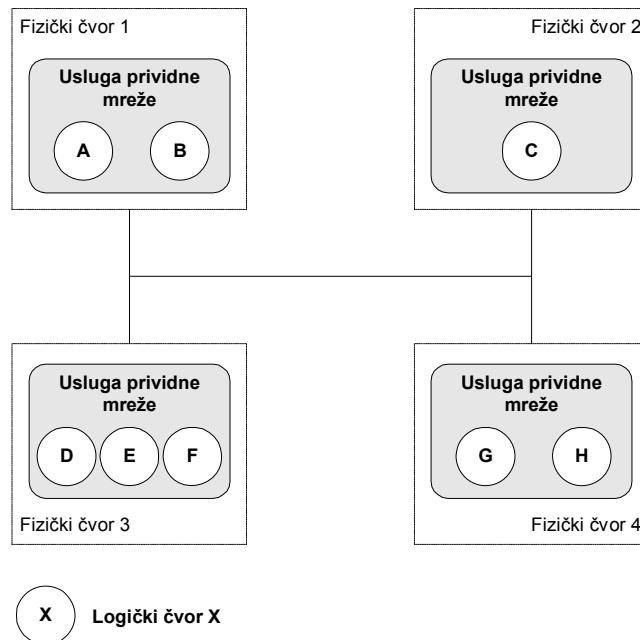
Slika 6.9 Model sigurnog prijenosa podataka u sustavu prividne mreže podijeljen na elemente sigurnosti od točke do točke i elemente sigurnosti od kraja do kraja

Poruke koje se prenose logičkim komunikacijskim prostorom prividne mreže, a služe za pozivanje programskih usluga, sastoje se od tijela i zaglavlja. U tijelu poruke nalaze se primjenske informacije koje pozivajuća programska usluga prenosi pozvanoj usluzi. Zaglavljje poruke sadrži informacije o izvořnom i odredišnom logičkom čvoru te čvorovima usmjernicima. Podaci sadržani u tijelu poruke prenose se u neizmijenjenom obliku od izvořnog do odredišnog logičkog čvora. Podaci u zaglavljiju poruke dijele se na promjenjive i nepromjenjive. Cijelim putem od izvořnog do odredišnog logičkog čvora, putem svih čvorova usmjernika, u nepromjenjenom obliku se prenose informacije o izvořnom i odredišnom logičkom čvoru. Informacije o čvorovima usmjernicima mijenjaju se na svakom čvoru usmjerniku, budući da su njihove vrijednosti različite na pojedinim dijelovima komunikacijskog puta između pojedinih parova čvorova usmjernika. Postojanost podataka sadržanih u tijelu poruke i podataka o izvořnom i odredišnom čvoru sadržanih u zaglavljiju poruke je, dakle, očuvana od kraja do kraja komunikacijskog puta. Ti podaci, stoga, podliježu provedbi postupaka sigurnosti od kraja do kraja. Postojanost podataka o čvorovima usmjernicima očuvana je samo na dijelu komunikacijskog puta između dviju točaka, odnosno dvaju čvorova usmjernika. Nad tim se podacima provode postupci sigurnosti od točke do točke.

Provjeda postupaka sigurnosti od kraja do kraja uključuje postupke kriptiranja i digitalnog potpisivanja podataka iz tijela poruke te digitalnog potpisivanja podataka o izvorišnom i odredišnom logičkom čvoru iz zaglavlja poruke. Kriptiranje podataka provodi se s ciljem očuvanja njihove tajnosti, dok se postupci digitalnog potpisivanja omogućavaju provjeru njihove vjerodostojnosti i izvornosti. Podaci su kriptirani javnim ključem odredišnog logičkog čvora te potpisani tajnim ključem izvorišnog čvora. Nad podacima iz zaglavlja poruke koji sadrže informacije čvorovima usmjernicima provode se postupci sigurnosti od točke do točke. Između svaka dva čvora usmjernika, ti se podaci digitalno potpisuju tajnim ključem izvorišnog čvora usmjernika.

7 Arhitektura prividne mreže računalnih sustava zasnovanih na uslugama

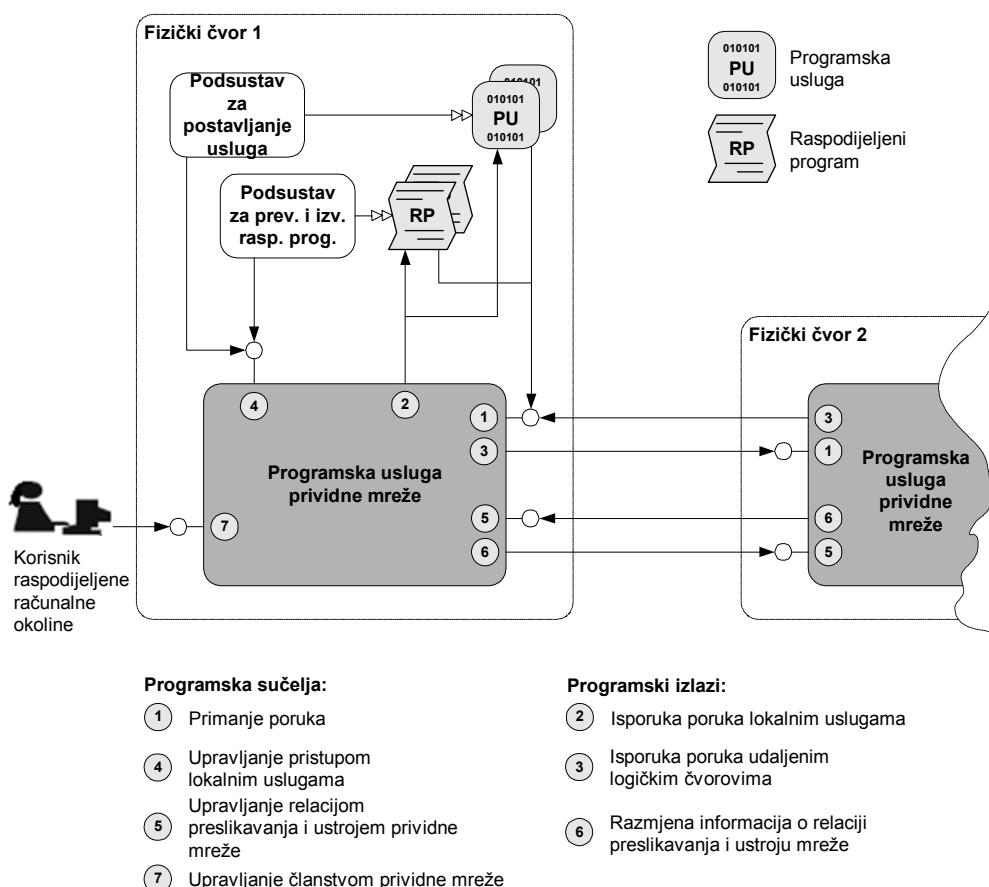
Prividna mreža računalnih sustava zasnovanih na uslugama je raspodijeljeni programski sustav čiji se elementi nalaze na svim čvorovima fizičke mreže kojima su pridruženi logički čvorovi prividne mreže. Programski sustav prividne mreže čini skup programskih usluga. Programska usluga prividne mreže (engl. *overlay service*) čini jedinstvenu i samostalnu programsku cjelinu koja obavlja funkciju sustava prividne mreže na jednom čvoru fizičke mreže. Rad cijelokupnog raspodijeljenog sustava prividne mreže postiže se postavljanjem programske usluge prividne mreže na svaki čvor fizičke mreže koji sudjeluje u oblikovanju prividne mreže. Raspodijeljeni programski sustav prividne mreže prikazan je slijekom 7.1. Prikazani računalni sustav sastoji se od četiri fizička čvora povezana računalnom mrežom. Na svakom čvoru fizičke mreže postavljena je programska usluga prividne mreže. Programska usluga prividne mreže omogućava postavljanje proizvoljnog broja logičkih čvorova na svaki čvor fizičke mreže. U prikazanom primjeru, na fizičkim čvorovima 1 i 4 postavljena su po dva logička čvora, na fizičkom čvoru 2 postavljen je jedan logički čvor, dok su na fizičkom čvoru 3 postavljena tri logička čvora.



Slika 7.1 Raspodijeljeni programski sustav prividne mreže

Programska usluga prividne mreže prema ostalim dijelovima prividne raspodijeljene računalne okoline izlaže četiri ulazna programska sučelja. Povezanost programske usluge prividne mreže s ostalim dijelovima prividne raspodijeljene računalne okoline prikazana je

slikom 7.2. Putem ulaznih programskih sučelja, ostali dijelovi prividne raspodijeljene računalne okoline uspostavljaju vezu s uslugom prividne mreže. Istovremeno, usluga prividne mreže komunikaciju s njima uspostavlja koristeći tri programska izlaza. Programski izlazi koriste se za prijenos podatkovnih i upravljačkih poruka prema ostalim programskim uslugama na lokalnom ili udaljenim logičkim čvorovima. Dva programska sučelja usluge prividne mreže dostupna su s udaljenih logičkih čvorova, dok su preostala dva namijenjena lokalnoj uporabi.

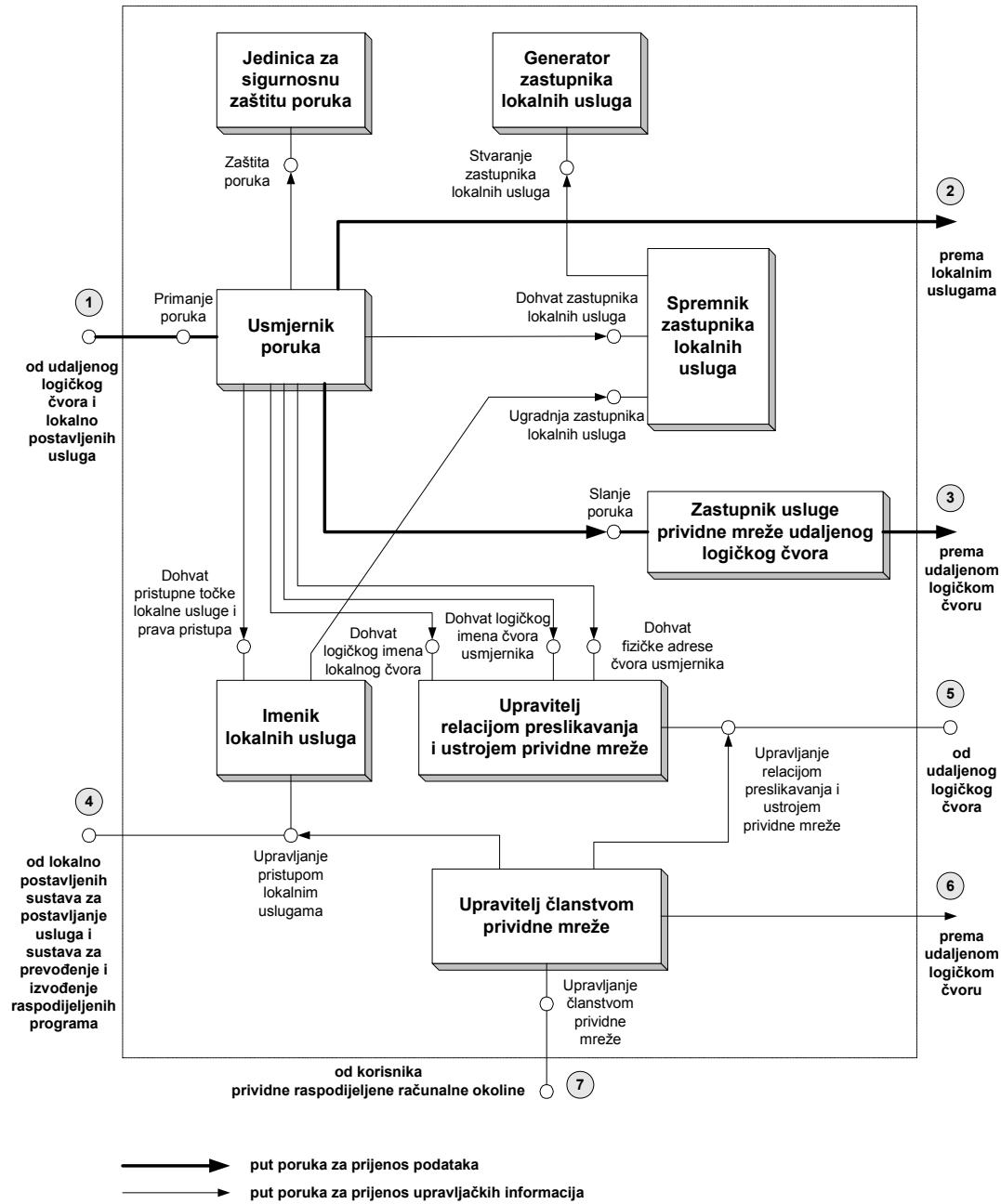


Slika 7.2 Povezanost programske usluge prividne mreže s ostalim dijelovima prividne raspodijeljene računalne okoline

Sučelje za primanje poruka (1) koristi se za razmjenu poruka putem logičkih komunikacijskih veza među logičkim čvorovima. Putem sučelja za primanje poruka, usluga prividne mreže prima poruke od programskih usluga smještenih na lokalnom logičkom čvoru, kao i od usluga prividne mreže udaljenih logičkih čvorova. Programske usluge smještene na lokalnom logičkom čvoru koje koriste sučelje za primanje poruka javljaju se u obliku primjenskih programskih usluga, usluga za suradnju i natjecanje te u obliku raspodijeljenih programa. Poruke zaprimljene na sučelju za primanje poruka isporučuju se

putem dvaju programskih izlaza. Programski izlaz (2) koristi se za slanje poruka programskim uslugama i raspodijeljenim programima smještenima na lokalnom čvoru prividne mreže. Programski izlaz (3) koristi se za slanje poruka programskim uslugama i raspodijeljenim programima smještenima na udaljenim logičkim čvorovima. Programsko sučelje za upravljanje pristupom lokalnim uslugama (4) koristi podsustav za postavljanje usluga tijekom postavljanja usluga, odnosno podsustav za prevođenje i izvođenje raspodijeljenih programa tijekom postavljanja raspodijeljenih programa na lokalni logički čvor. Putem tog sučelja, novopostavljene usluge i raspodijeljeni programi se prijavljuju u sustav prividne mreže te tako postaju dostupni u njezinom logičkom komunikacijskom prostoru. Tijekom promjena u obliku i ustroju prividne mreže koje nastupaju pri ulasku novog čvora u sustav prividne mreže ili pri napuštanju mreže od strane postojećeg logičkog čvora koristi se sučelje za upravljanje relacijom preslikavanja i ustrojem prividne mreže (5) i programski izlaz za razmjenu informacija o relaciji preslikavanja i ustroju mreže (6). Programsko sučelje za upravljanje relacijom preslikavanja i ustrojem prividne mreže koristi se za primanje informacija o nastalim promjenama od udaljenih logičkih čvorova, dok se programskim izlazom za razmjenu informacija o relaciji preslikavanja i ustroju mreže takve promjene dojavljuju udaljenim logičkim čvorovima. Korisniku prividne raspodijeljene računalne okoline izloženo je programsko sučelje za upravljanje članstvom prividne mreže (7). Koristeći to programsko sučelje, korisnik postavlja zahtjeve za uključivanjem novog i isključivanjem postojećeg logičkog čvora iz sustava prividne mreže.

Programska usluga prividne mreže sastoji se od osam programskih komponenata. Arhitektura programske usluge prividne mreže prikazana je slikom 7.3. Programska usluga prividne mreže sastoji se od *Usmjernika poruka*, *Imenika lokalnih usluga*, *Spremnika zastupnika lokalnih usluga*, *Generatora zastupnika lokalnih usluga*, *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*, *Zastupnika usluge prividne mreže udaljenog logičkog čvora*, *Jedinice za sigurnosnu zaštitu poruka* i *Upravitelja članstvom prividne mreže*. Arhitekturom programske usluge prividne mreže prikazanom slikom 7.3 omogućeno je postavljanje proizvoljnog broja nezavisnih logičkih čvorova na jedan fizički čvor. Za sve logičke čvorove postavljene na jednom fizičkom čvoru postoji samo jedna programska usluga prividne mreže. Razdvajanje i razlikovanje logičkih čvorova, kao apstraktnih programskih tvorevina, ostvareno je stukturama podataka unutar te usluge. Svakom logičkom čvoru pridružene su zasebne strukture podataka. Povezanost komponenata usluge prividne mreže prikazana je slikom 7.3, dok su arhitektura, funkcija, načelo rada i strukture podataka svake pojedine komponente opisane u nastavku ovog poglavlja.



Slika 7.3 Arhitektura programske usluge prividne mreže

7.1 Usmjernik poruka

Usmjernik poruka je središnja komponenta programske usluge prividne mreže. Uloga *Usmjernika poruka* je prihvatanje i proslijeđivanje poruka u skladu s definiranim pravilima usmjeravanja. *Usmjernik poruka* prihvata poruke od primjenskih programskih usluga smještenih na lokalnom logičkom čvoru te od udaljenih logičkih čvorova, odnosno od

programskih usluga prividne mreže smještenih na udaljenim logičkim čvorovima. Isto tako, primljene poruke proslijeduje primjenskim programskim uslugama smještenima na lokalnom logičkom čvoru te programskim uslugama prividne mreže smještenima na udaljenim logičkim čvorovima. Ovisno o vrsti pošiljatelja i primatelja od kojih *Usmjernik poruka* prima, odnosno kojima šalje poruke, određuje se uloga logičkog čvora. Tablicom 7.1 prikazane su moguće uloge koje poprima logički čvor u određenim okolnostima.

Tablica 7.1 Uloga logičkog čvora u komunikaciji u ovisnosti o vrsti pošiljatelja i primatelja poruke

Vrsta pošiljatelja poruke	Vrsta primatelja poruke	Uloga logičkog čvora
Lokalna primjenska programska usluga	Lokalna primjenska programska usluga	Izvorišni i odredišni čvor
Lokalna primjenska programska usluga	Programska usluga prividne mreže	Izvorišni čvor
Programska usluga prividne mreže	Lokalna primjenska programska usluga	Odredišni čvor
Programska usluga prividne mreže	Programska usluga prividne mreže	Čvor usmjernik

Ako *Usmjernik poruka* zaprili poruku od lokalne primjenske programske usluge, onda logički čvor poprima ulogu izvorišnog logičkog čvora. Ako je primljenu poruku potrebno isporučiti lokalnoj primjenskoj programskoj usluzi, onda lokalni logički čvor poprima ulogu odredišnog logičkog čvora. Ako su pošiljatelj i primatelj poruke primjenske programske usluge smještene na istom logičkom čvoru, onda logički čvor istovremeno poprima ulogu izvorišnog i odredišnog logičkog čvora. Ako primjenska programska usluga smještena na lokalnom logičkom čvoru poziva primjensku uslugu smještenu na udaljenom logičkom čvoru, *Usmjernik poruka* lokalnog logičkog čvora tu poruku treba proslijediti na sučelje za primanje poruka programske usluge prividne mreže smještene na udaljenom logičkom čvoru. Lokalni logički čvor u tom slučaju poprima ulogu izvorišnog logičkog čvora. Ako *Usmjernik poruka* zaprili poruku pristiglu od udaljenog logičkog čvora, odnosno od programske usluge prividne mreže smještene na udaljenom logičkom čvoru, onda je primjenska programska usluga koja je šalje smještena na udaljenom logičkom čvoru. Uloga lokalnog logičkog čvora određuje se na osnovi primatelja poruke. Ako je poruku potrebno isporučiti lokalnoj primjenskoj usluzi, lokalni čvor poprima ulogu odredišnog logičkog čvora. Ako je poruku potrebno proslijediti drugom udaljenom logičkom čvoru, odnosno programskoj usluzi prividne mreže udaljenog logičkog čvora, lokalni čvor poprima ulogu usmjernika.

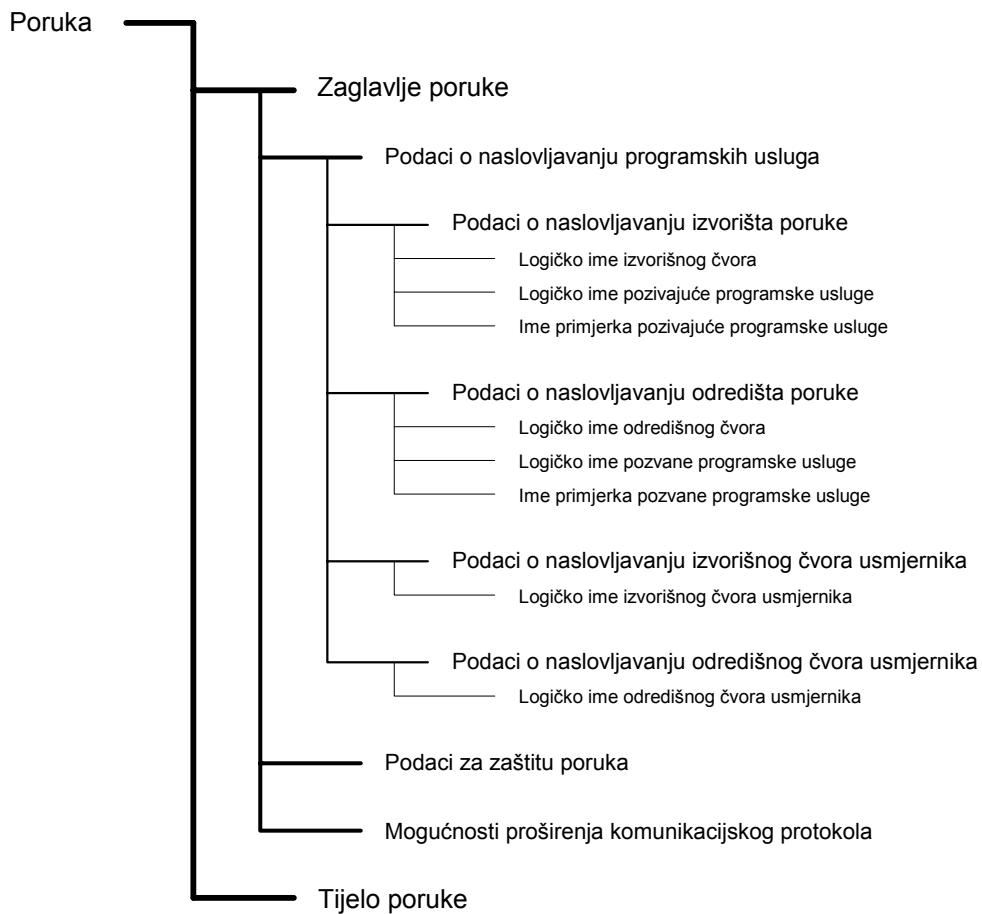
7.1.1 Struktura poruke za ostvarivanje komunikacije logičkih čvorova

Poruke koje razmjenjuju logički čvorovi oblikovane su u skladu s odredbama *SOAP* protokola. Poruke oblikovane prema odredbama *SOAP* protokola sastoje se od dva osnovna dijela. Prvi dio poruke čini zaglavljje poruke, a drugi dio je tijelo poruke. Zaglavljje poruke sadrži upravljačke podatke, potrebne za slanje poruke od pozivajuće do pozvane programske usluge te za obradu tijela poruke. Tijelo poruke sadrži primjenske podatke koji se prenose pri pozivu programskih usluga, odnosno pri vraćanju odgovora s rezultatima izvođenja programskih usluga. Primjerice, pri pozivu programske usluge, tijelo poruke sadrži ime operacije koja se poziva na odredišnoj programskoj usluzi te imena i vrijednosti parametara poziva.

U postupku usmjeravanja poruka koriste se podaci sadržani u zaglavljima poruka. Zaglavljje poruke je neobvezna cjelina *SOAP* protokola. Podaci sadržani u zaglavljiju koriste se za proširivanje osnovnog oblika *SOAP* protokola različitim primjenskim protokolima. Prividna mreža računalnih sustava zasnovanih na uslugama proširuje osnovni *SOAP* protokol zaglavljima za ostvarivanje funkcionalnosti *WS-Addressing* i *WS-Security* primjenskih protokola. Zaglavljja *WS-Addressing* protokola koriste se za naslovljavanje programskih usluga i za usmjeravanje poruka među logičkim čvorovima. Zaglavljja *WS-Security* protokola koriste se za sigurnosnu zaštitu poruka pri prijenosu računalnom mrežom. Osim podataka za ostvarivanje funkcionalnosti ovih dvaju primjenskih protokola, zaglavljje poruke je proširiva cjelina koju je moguće nadopuniti dodatnim podacima u slučaju potrebe za uvođenjem novog primjenskog protokola.

Struktura *SOAP* poruke s istaknutim dijelovima zaglavja značajnima za postupak usmjeravanja prikazana je slikom 7.4. *Usmjernik poruka* obavlja usmjeravanje poruka ispitujući dio zaglavja s podacima o naslovljavanju programskih usluga. Podaci o naslovljavanju programskih usluga podijeljeni su u četiri cjeline. Prvu cjelinu čine podaci o naslovljavanju izvorišta poruka, a drugu podaci o naslovljavanju odredišta poruke. Vrijednosti podataka o naslovljavanju izvorišta i odredišta poruka definiraju se na izvorišnom logičkom čvoru prividne mreže od strane pozivajuće programske usluge. Za vrijeme prolaska poruke kroz mrežu logičkih čvorova njihove se vrijednosti ne mijenjaju te oni imaju obilježja postojanosti od kraja do kraja (engl. *end to end*). Obje cjeline sadrže podatke trorazinskog sustava naslovljavanja usluga koji na prvoj razini sadrži logičko ime izvorišnog, odnosno odredišnog logičkog čvora, na drugoj razini logičko ime pozivajuće, odnosno pozvane programske usluge te na trećoj razini ime pozivajućeg, odnosno pozvanog primjerka usluge. Treća i četvrta cjelina sadrže podatke o čvorovima usmjernicima na putu

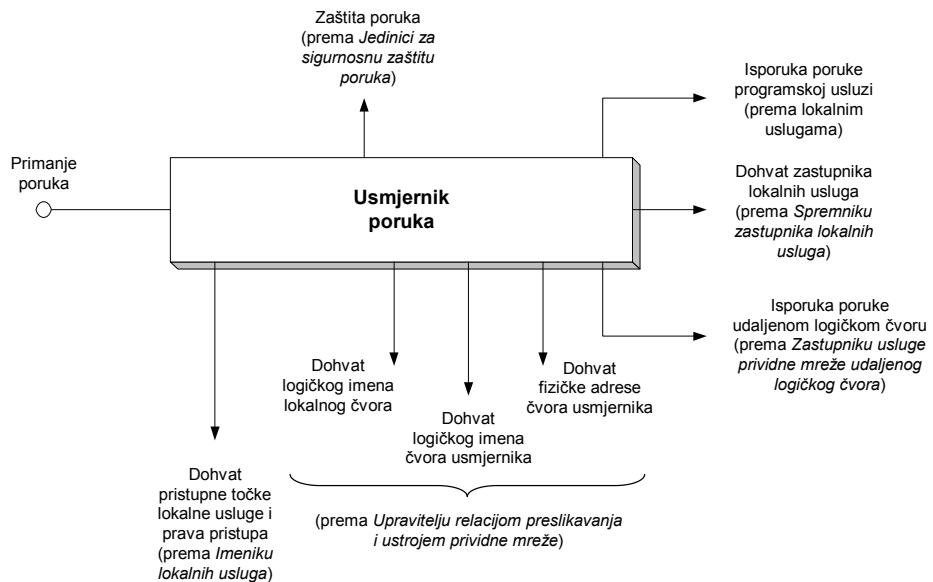
od izvořišnog do odredišnog logičkog čvora. Podaci o čvorovima usmjernicima mijenjaju se na svakom dijelu komunikacijskog puta pri prolasku poruke od izvořišnog do odredišnog logičkog čvora te oni imaju obilježja postojanosti od točke o točke (engl. *point to point*). Svaki čvor usmjernik u primljenoj poruci u dijelu s podacima o naslovljavanju izvořišnog čvora usmjernika pronalazi ime logičkog čvora koji na komunikacijskom putu poruke njemu neposredno prethodi. U dijelu s podacima o naslovljavanju odredišnog čvora usmjernika, čvor usmjernik pronalazi vlastito logičko ime. U poruci koju posljeđuje sljedećem čvoru usmjerniku na komunikacijskom putu, logičko ime izvořišnog čvora usmjernika zamjenjuje vlastitim logičkim imenom. Logičko ime odredišnog čvora usmjernika nadomešta logičkim imenom čvora kojemu prosljeđuje poruku. Logičko ime odredišnog čvora usmjernika na koji se usmjerava primljena poruka *Usmjernik poruka* dohvaca od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*.



Slika 7.4 Struktura *SOAP* poruke za ostvarivanje komunikacije među logičkim čvorovima

7.1.2 Arhitektura usmjernika poruka

Usmjernik poruka je ulazna komponenta programske usluge prividne mreže. Arhitektura *Usmjernika poruka* prikazana je slikom 7.5. Prema ostalim uslugama raspodijeljenog računalnog sustava, *Usmjernik poruka* izlaže sučelje za primanje poruka. Putem tog sučelja, *Usmjernik poruka* prihvata ulazne poruke usmjerene prema lokalnom logičkom čvoru. Pri obavljanju postupka usmjeravanja poruka, *Usmjernik poruka* koristi funkcionalnosti ostalih komponenata usluge prividne mreže s kojima je povezan svojim programskim izlazima. Sučelju za zaštitu poruka *Jedinice za sigurnosnu zaštitu poruka Umjernik poruka* pristupa kako bi se nad ulaznim porukama proveli postupci kriptiranja i digitalnog potpisivanja, a nad izlaznim porukama postupci dekriptiranja i provjere valjanosti digitalnih potpisa. Dohvat logičkog imena lokalnog logičkog čvora obavlja se pristupanjem sučelju za dohvat logičkog imena lokalnog čvora pri *Upravitelju relacijom preslikavanja i ustrojem prividne mreže*. Tijekom isporuke poruke programskoj usluzi smještenoj na lokalnom logičkom čvoru, čvor usmjernik koristi funkcionalnosti *Imenika lokalnih usluga* putem sučelja za dohvat pristupne točke usluge i njezinih prava pristupa te funkcionalnosti *Spremnika zastupnika lokalnih usluga* putem sučelja za dohvat zastupnika lokalnih usluga. Za isporuku poruke udaljenom logičkom čvoru, pristupa se sučeljima za dohvat logičkog imena i fizičke adrese čvora usmjernika *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*. Slanje poruke računalnom mrežom do udaljenog logičkog čvora obavlja se pristupanjem sučelju za slanje poruka *Zastupnika usluge prividne mreže udaljenog logičkog čvora*.



Slika 7.5 Arhitektura *Usmjernika poruka*

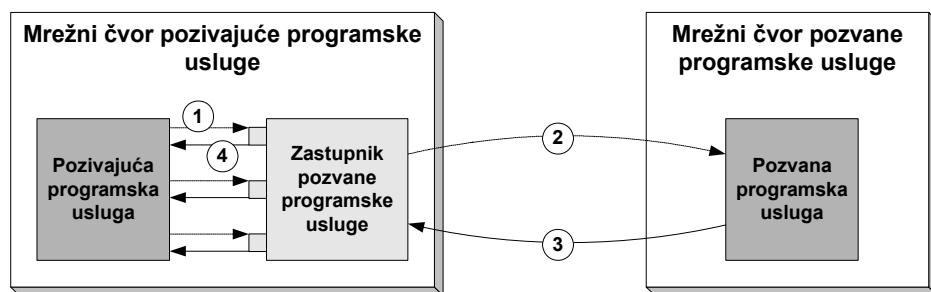
Svi programski izlazi kojima je *Usmjernik poruka* povezan s ostalim komponentama usluge prividne mreže koriste se u postupku prosljeđivanja primljenih poruka na odgovarajuća odredišta. Odluku o odredištu kojemu je potrebno proslijediti poruku, *Usmjernik poruka* donosi na osnovi podataka o naslovljavanju programskih usluga sadržanih u primljenim porukama. Za svaku primljenu poruku, *Usmjernik poruka* ispituje logičko ime odredišnog logičkog čvora. Usaporedbom imena odredišnog logičkog čvora s imenom lokalnog logičkog čvora, *Usmjernik poruka* donosi odluku o odredištu poruke. Ako su imena lokalnog i odredišnog logičkog čvora jednaka, poruku je potrebno proslijediti programskoj usluzi smještenoj na lokalnom logičkom čvoru. Utvrdi li se razlika u imenima, poruku je potrebno proslijediti udaljenom logičkom čvoru.

7.1.3 Isporuka poruke lokalnim programskim uslugama

Tijekom isporuke poruke programskoj usluzi smještenoj na lokalnom logičkom čvoru, *Usmjernik poruka* koristi podatke druge razine naslovljavanja usluga. Iz primljene poruke dohvaća se logičko ime pozvane usluge. Za fizičku isporuku poruke programskoj usluzi, *Usmjernik poruka* mora utvrditi adresu pristupne točke te usluge na lokalnom čvoru fizičke mreže. Primjer pristupne točke programske usluge je njezina jedinstvena oznaka sredstva u obliku

`http://localhost/ProgramskeUsluge/UslugaBanke/Banka.asmx`

Budući da se na drugoj razini naslovljavanja usluga koriste logička imena usluga, prije upućivanja poziva usluzi potrebno je njezino logičko ime nadomjestiti njezinom fizičkom pristupnom točkom.



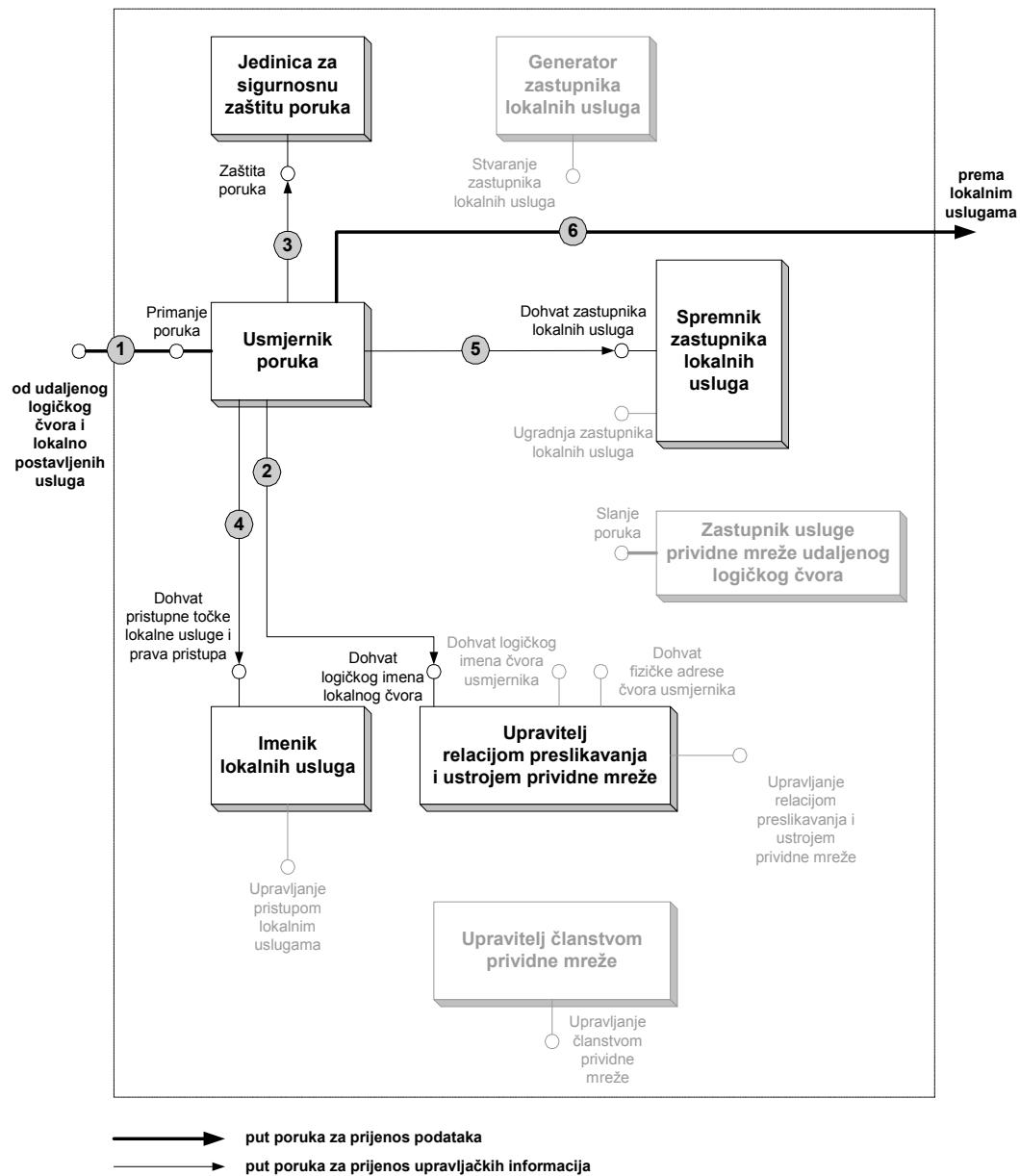
Slika 7.6 Opći model pozivanja programske usluge

Osim pristupne točke, za ostvarivanje pristupa programskoj usluzi potrebno je izgraditi odgovarajući zastupnik programske usluge (engl. *service proxy*). Zastupnik programske usluge je prividna programska usluga smještena na strani pozivatelja. Uloga zastupnika programske usluge je pružanje programskog sučelja za poziv usluge pozivajućim

programskim uslugama i ostalim pozivajućim programima te prijenos poruka sa zahtjevima i odgovorima fizičkom mrežom između pozivajuće i pozvane programske usluge. Model poziva programske usluge prikazan je slikom 7.6. Pozivajuća programska usluga koristi sučelje zastupnika pozvane usluge za slanje zahtjeva pozvanoj usluzi (1). Zastupnik pozvane usluge primljeni zahtjev ugrađuje u poruku fizičkog prijenosnog protokola i prenosi ga računalnom mrežom do pozvane usluge (2). Nakon obrade zahtjeva, pozvana usluga vraća rezultate obrade svom zastupniku smještenom na strani pozivajuće usluge u obliku odgovora ugrađenog u poruku fizičkog prijenosnog protokola (3). Zastupnik pozvane usluge iz primljene poruke dohvaća odgovor na postavljeni zahtjev i prosljeđuje ga pozivajućoj usluzi (4).

Model pozivanja programske usluge prikazan slikom 7.6 koristi se za isporuku poruke lokalnim programskim uslugama. Isporuka poruke odvija se u šest koraka, čiji je slijed prikazan slikom 7.7. Pri postupku isporuke poruke lokalnoj programskoj usluzi sudjeluje pet od ukupno osam komponenata programske usluge prividne mreže. U prvom koraku *Usmjernik poruka* zaprima poruku na svom sučelju za primanje poruka. Izvorište poruke može biti programska usluga smještena na lokalnom logičkom čvoru ili programska usluga prividne mreže udaljenog logičkog čvora. Nakon primanja poruke, *Usmjernik poruka* pokreće postupak usporedbe imena lokalnog logičkog čvora s imenom odredišnog logičkog čvora. Ime odredišnog logičkog čvora dohvaća se iz podataka o naslovljavanju pozvane programske usluge koji su sadržani u primljenoj poruci. Ime lokalnog logičkog čvora *Usmjernik poruka* dohvaća u drugom koraku postupka od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*, pristupajući njegovom sučelju za dohvati imena lokalnog logičkog čvora. Na osnovi rezultata usporedbe imena lokalnog i odredišnog logičkog čvora odlučuje se o poduzimanju dalnjih koraka. Za nastavak postupka isporuke poruke programskoj usluzi smještenoj na lokalnom logičkom čvoru imena lokalnog i odredišnog logičkog čvora moraju biti podudarna.

Ako je komunikacija među logičkim čvorovima zaštićena sigurnosnim mjerama prijenosa podataka, prije daljnje obrade poruke potrebno je provesti odgovarajuće sigurnosne zahvate nad sadržajem poruke. Budući da je lokalni logički čvor u ovom slučaju krajnje odredište primljene poruke, prije njezine isporuke potrebno je provesti dekriptiranje sadržaja tijela poruke te provjeriti ispravnost digitalnih potpisa izvorišnog logičkog čvora i čvora usmjernika od kojeg je poruka primljena. Provedba sigurnosnih operacija nad sadržajem poruke obavlja se pristupanjem sučelju za zaštitu poruka *Jedinice za sigurnosnu zaštitu poruka*. Postupci kriptiranja i dekriptiranja sadržaja poruka, digitalnog potpisivanja poruka te provjere ispravnosti digitalnih potpisa podrobnije su opisani u odjeljku 6.9 i u [109].



Slika 7.7 Postupak isporuke poruke programskoj usluzi smještenoj na lokalnom logičkom čvoru

Prije isporuke poruke pozvanoj programskoj usluzi, *Usmjernik poruka* provjerava da li pozivajuća programska usluga ima odgovarajuća prava pristupa za poziv željene usluge. Ako pozivajuća usluga udovoljava postavljenim ograničenjima koja su definirana pravima pristupa, *Usmjernik poruka* nadomešta logičko ime pozvane programske usluge njezinom fizičkom pristupnom točkom. Provjera prava pristupa i pribavljanje pristupne točke do pozvane programske usluge obavlja se pristupanjem *Imeniku lokalnih usluga* putem sučelja za dohvat pristupne točke lokalne usluge i prava pristupa u četvrtom koraku postupka isporuke poruke.

U petom koraku postupka isporuke poruke *Usmjerenik poruka* pokreće postupak pribavljanja zastupnika lokalne usluge koji omogućava pristup do pozvane usluge. Zastupnici lokalnih usluga spremjeni su u *Spremniku zastupnika lokalnih usluga* i dostupni su putem sučelja za dohvati zastupnika lokalnih usluga. Nakon što je pribavio zastupnik odgovarajuće lokalne usluge, *Usmjerenik poruka* u šestom koraku isporučuje poruku pozvanoj programskoj usluzi na lokalnom logičkom čvoru. Pribavljanje odgovora od pozvane programske usluge također je uključeno u završni korak postupka isporuke poruke. Prosljeđivanje odgovora pozivajućoj programskoj usluzi od *Usmjernika poruka* odredišnog logičkog čvora do *Usmjernika poruka* izvođenog logičkog čvora slijedi isto načelo koje vrijedi i za prosljeđivanje zahtjeva.

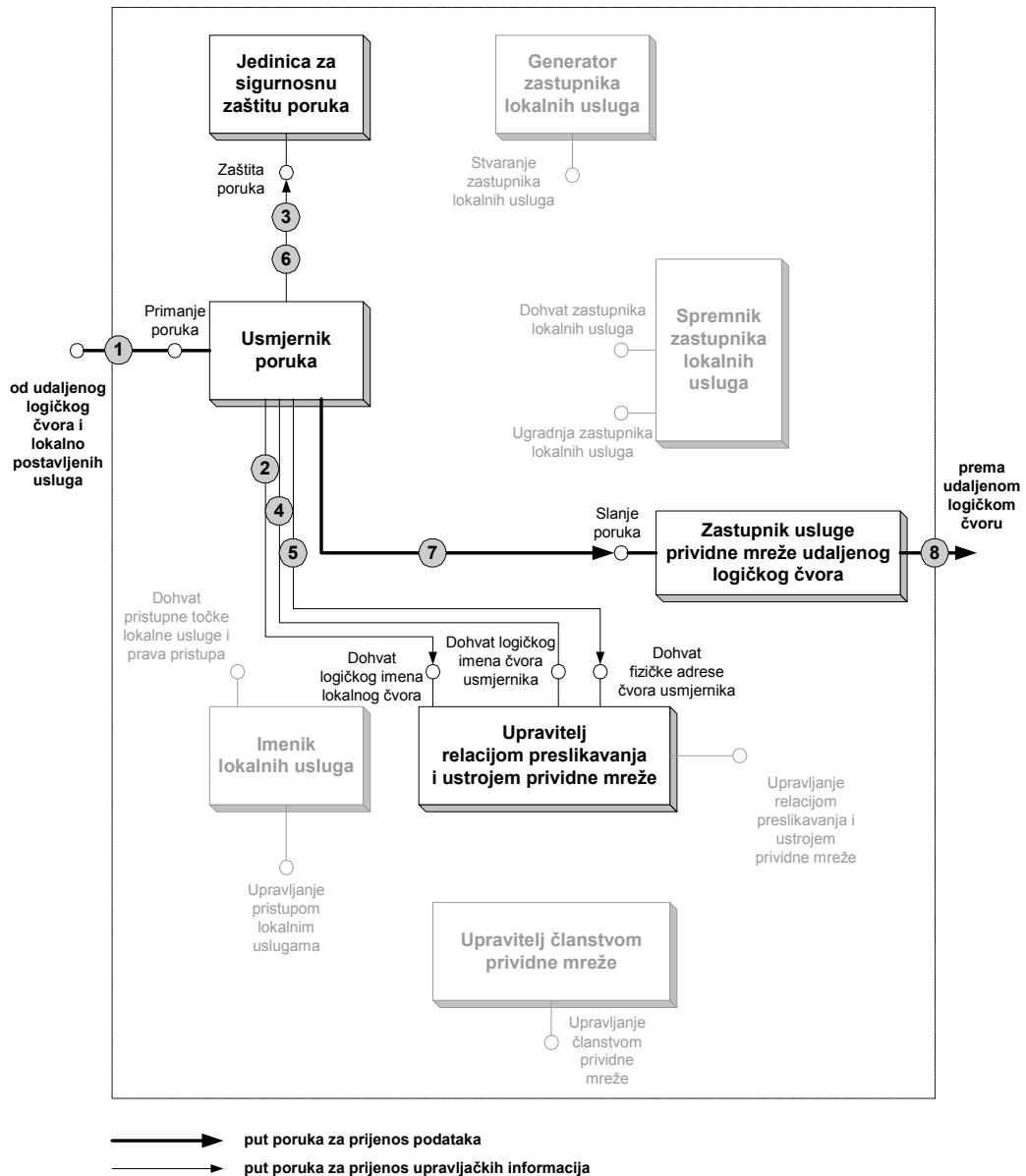
7.1.4 Isporuka poruke udaljenim logičkim čvorovima

U slučajevima kada je utvrđena nepodudarnost imena lokalnog i odredišnog logičkog čvora, *Usmjerenik poruka* započinje postupak isporuke poruke udaljenom logičkom čvoru. Taj se postupak sastoji od osam koraka u slučaju primjene zaštitnih mjera sigurnog prijenosa podataka u komunikaciji logičkih čvorova, odnosno u šest koraka ako komunikacija nezaštićenim porukama predstavlja zadovoljavajuće rješenje. Postupak isporuke poruke udaljenom logičkom čvoru prikazan je slikom 7.8. Od ukupno osam programskih komponenata koje čine programsku uslugu prividne mreže, ovaj postupak zahtijeva međudjelovanje njih četiri. U postupak su uključeni *Usmjerenik poruka*, *Jedinica za sigurnosnu zaštitu poruka*, *Upravitelj relacijom preslikavanja i ustrojem prividne mreže* te *Zastupnik usluge prividne mreže udaljenog logičkog čvora*.

Prva dva koraka u postupku isporuke poruke udaljenom logičkom čvoru jednaka su kao i u postupku isporuke poruke programskoj usluzi smještenoj na lokalnom logičkom čvoru. U prvom koraku *usmjerenik poruka* na svojem sučelju za primanje poruka zaprima poruku od lokalne programske usluge ili udaljenog logičkog čvora. U drugom koraku *Usmjerenik poruka* od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* dohvaca ime lokalnog logičkog čvora, kako bi ga usporedio s imenom odredišnog logičkog čvora. Za nastavak postupka isporuke poruke udaljenom logičkom čvoru potrebno je utvrditi nepodudarnost tih dvaju logičkih imena. Nepodudarnost logičkih imena lokalnog i odredišnog logičkog čvora je naznaka da poruka nije namjenjena programskoj usluzi smještenoj na lokalnom logičkom čvoru, već ju je potrebno proslijediti sljedećem logičkom čvoru na komunikacijskom putu od izvođenog do odredišnog logičkog čvora.

Ako se komunikacija odvija primjenom zaštitnih mehanizama sigurnog prijenosa podataka, *Usmjerenik poruka* pristupa *Jedinici za sigurnosnu zaštitu poruka*. U primljenoj

poruci potrebno je provesti postupak provjere ispravnosti digitalnih potpisa. Poruku nije potrebno niti ju je moguće dekriptirati na lokalnom logičkom čvoru, budući da ona njemu nije namenjena te on ne raspolaže potrebnim ključevima za obavljanje postupka dekriptiranja.



Slika 7.8 Postupak isporuke poruke udaljenom logičkom čvoru

Ako je u postupku provjere valjanosti digitalnih potpisa utvrđena njihova ispravnost, *Usmjernik poruka* pokreće postupak prikupljanja podataka za usmjeravanje poruke do sljedećeg čvora usmjernika na komunikacijskom putu od izvorišnog do odredišnog logičkog čvora. Postupak započinje pribavljanjem logičkog imena sljedećeg čvora usmjernika.

Logičko ime čvora usmjernika dohvaća se od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* u četvrtom koraku, pristupajući njegovom sučelju za dohvat logičkog imena čvora usmjernika. *Upravitelj relacijom preslikavanja i ustrojem prividne mreže* traženo logičko ime dohvaća iz svojih tablica usmjeravanja. Nakon što je pribavljen potrebno logičko ime, za postupak upućivanja poruke do čvora usmjernika zadanog logičkog imena potrebno je poznavati adresu fizičkog čvora na kojemu se taj logički čvor nalazi. Adresa fizičkog čvora određena je relacijom preslikavanja prividne na fizičku mrežu, čije podatke održava *Upravitelj relacijom preslikavanja i ustrojem prividne mreže*. Pristupanjem njegovom sučelju za dohvat fizičke adrese čvora usmjernika u petom koraku, *Usmjernik poruka* dohvaća fizičku adresu mrežnog čvora na koji je potrebno proslijediti primljenu poruku.

Nakon što je pribavio podatke o sljedećem čvoru usmjerniku na putu poruke od izvorišnog do odredišnog logičkog čvora, *Usmjernik poruka* nadomješta potrebne informacije u zaglavlju poruke koje se odnose na imena logičkih čvorova usmjernika. U podacima o izvorišnom čvoru usmjerniku, ime logičkog čvora od kojeg je poruka pristigla nadomješta se imenom lokalnog logičkog čvora. U podacima o odredišnom čvoru usmjerniku, ime lokalnog logičkog čvora nadomješta se imenom sljedećeg logičkog čvora usmjernika. Tako oblikovana poruka spremna je za isporuku sljedećem čvoru usmjerniku. Za osiguravanje sigurnosti njezina prijenosa mrežom, *Usmjernik poruka* u šestom koraku surađuje s *Jedinicom za sigurnosnu zaštitu poruka* kako bi se izmjenjeni podaci u zaglavlju poruke ponovno digitalno potpisali.

Nakon što je poruka pravilno oblikovana te su poduzete odgovarajuće sigurnosne mjere za zaštitu njezina sadržaja, pokreće se postupak fizičke isporuke poruke sljedećem čvoru usmjerniku. Isporuku poruka udaljenom logičkom čvoru *Usmjernik poruka* obavlja putem *Zastupnika usluge prividne mreže udaljenog logičkog čvora*. Putem tog zastupnika, fizički čvor na kojem je postavljen lokalni logički čvor šalje poruku fizičkim komunikacijskim vezama do fizičkog čvora na kojem je smješten udaljeni logički čvor. *Usmjernik poruka* u sedmom koraku postupka isporuke poruke *Zastupniku usluge prividne mreže udaljenog logičkog čvora* proslijeđuje poruku i adresu fizičkog čvora kojemu je tu poruku potrebno uputiti. *Zastupnik usluge prividne mreže udaljenog logičkog čvora* u završnom koraku postupka obavlja fizičku isporuku poruke fizičkom čvoru na kojem se nalazi naslovljeni logički čvor usmjernik. Usluga prividne mreže na udaljenom logičkom čvoru prima tu poruku na sučelju za primanje poruka vlastitog *Usmjernika poruka*. Ovisno o tome da li je naslovljeni logički čvor odredišni čvor ili samo još jedan u lancu čvorova

usmjernika, tamošnji *Usmjernik poruka* ponovno pokreće jedan od dva moguća postupka za daljnju isporuku primljene poruke.

7.2 Imenik lokalnih usluga

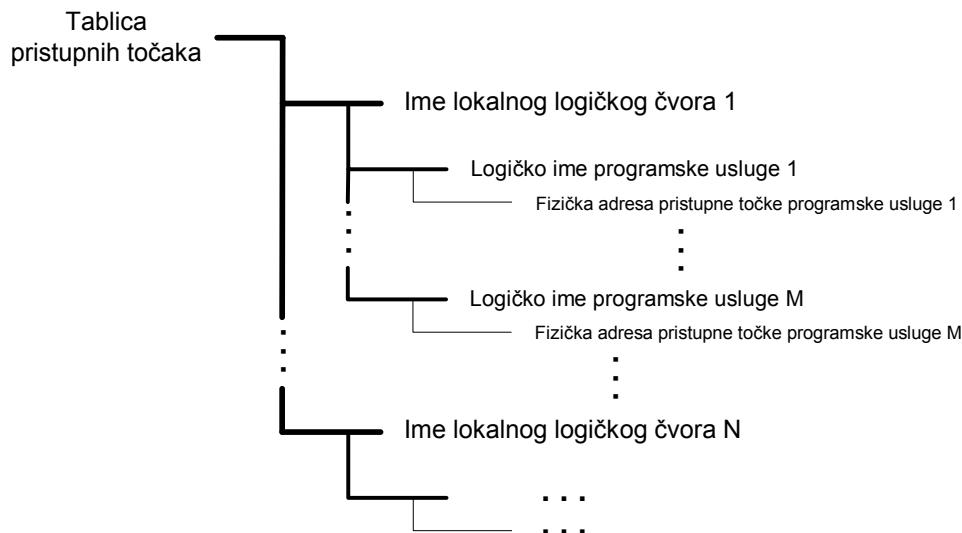
Imenik lokalnih usluga je programska komponenta usluge prividne mreže koja sadrži podatke o programskim uslugama postavljenim na lokalnom logičkom čvoru. Podatke sadržane u strukturama podataka *Imenika lokalnih usluga* koristi *Usmjernik poruka* u postupku isporuke poruke lokalnim programskim uslugama. *Imenik lokalnih usluga* sadrži dvije vrste podataka o programskim uslugama. Jedna vrsta podataka su podaci o pristupnim točkama programskih usluga na fizičkom čvoru mreže na kojem su smještene. Logička imena programskih usluga koja se koriste za naslovljavanje usluga u logičkom komunikacijskom prostoru prividne mreže zamjenjuju se njihovim fizičkim pristupnim točkama u trenutku isporuke poruke programskoj usluzi. Drugu vrstu podataka čine podaci o pravima pristupa do programskih usluga. Za svaku programsку uslugu postavljenu u logičkom komunikacijskom prostoru prividne mreže moguće je definirati pravila pristupa. Pravilima pristupa određuju se uvjeti pod kojima je moguće ostvariti pristup do funkcionalnosti usluge. *Imenik lokalnih usluga* za upravljanje podacima o pristupnim točkama programskih usluga i njihovim pravima pristupa koristi dvije strukture podataka. Podaci o pristupnim točkama lokalnih programskih usluga sadržani su u tablici pristupnih točaka. Podaci o pravima pristupa sadržani su u tablici prava pristupa.

7.2.1 Tablica pristupnih točaka

Tablica pristupnih točaka programskih usluga smještenih na lokalnom logičkom čvoru za svaku lokalno postavljenu programsku uslugu sadrži dvije vrijednosti. Jedna od vrijednosti predstavlja logičko ime lokalne programske usluge, a druga fizičku adresu njezine pristupne točke na lokalnom čvoru fizičke mreže. Struktura tablice pristupnih točaka lokalnih programskih usluga prikazana je slikom 7.9.

Kako na jednom čvoru fizičke mreže može biti postavljeno proizvoljno mnogo logičkih čvorova prividne mreže, a programska usluga prividne mreže je zajednička za sve logičke čvorove, tablica pristupnih točaka je podijeljena na nekoliko osnovnih cjelina. Svaka cjelina pripada jednom logičkom čvoru i sadrži podatke o pristupnim točkama programskih usluga koje su postavljene na tom logičkom čvoru. Ključ pretraživanja tablice pristupnih točaka je ime lokalnog logičkog čvora te logičko ime programske usluge za koju se traži adresa pristupne točke. Tablica pristupnih točaka je u radnom spremniku računala

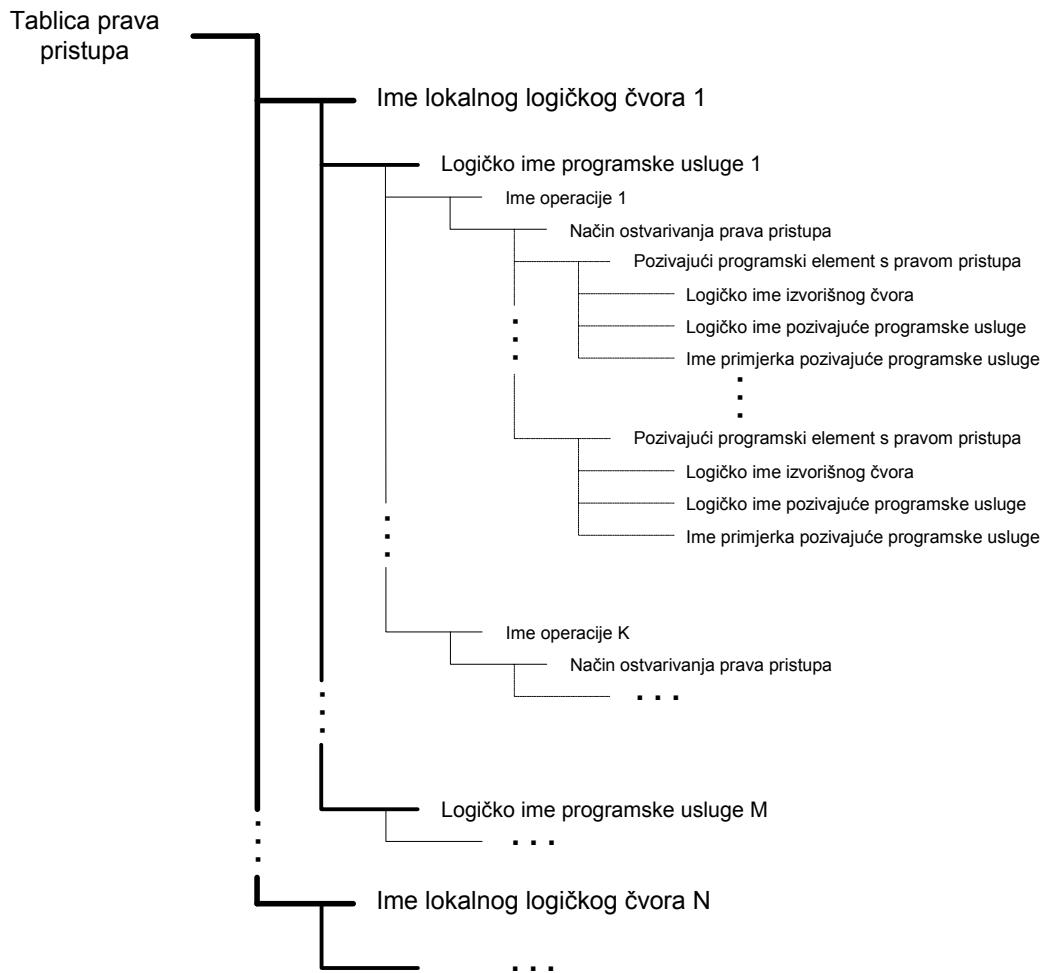
spremljena u obliku *XML* dokumenta. Njezina hijerarhijska struktura olakšava postupak pretraživanja tijekom dohvaćanja adrese pristupne točke programske usluge.



Slika 7.9 Struktura tablice pristupnih točaka lokalnih programskih usluga

7.2.2 Tablica prava pristupa

Prava pristupa programskim uslugama ispituju se na razini operacija pozvanih programskih usluga. Pozivajućoj programskoj usluzi moguće je dozvoliti pristup samo do određenih operacija pozvane usluge, dok joj je pristup ostalim operacijama moguće zabraniti. Za svaku operaciju pozvane programske usluge postoje tri moguća načina ostvarivanja prava pristupa: lokalni (engl. *local access*), udaljeni (engl. *remote access*) i udaljeni s ograničnjima (engl. *selective remote access*). Lokalnim pristupom dodjeljuju se prava pristupa do operacije pozvane programske usluge svim programskim uslugama koje su smještene na istom logičkom čvoru kao i pozvana usluga. Udaljenim pristupom dodjeljuju se prava pristupa do operacije pozvane programske usluge svim programskim uslugama koje su smještene na istom logičkom čvoru kao i pozvana usluga, kao i onima koje su smještene na udaljenim logičkim čvorovima. Drugim riječima, udaljeni pristup predstavlja operaciju programske usluge s javnim pravima pristupa koja ostvaruju sve programske usluge bez obzira na njihov položaj. Udaljeni pristup ne postavlja ograničenja na korištenje operacije programske usluge. Udaljeni pristup s ograničnjima je poopćena vrsta udaljenog pristupa. Udaljenim pristupom s ograničnjima dodjeljuju se prava pristupa do operacije pozvane programske usluge samo onim programskim uslugama koje su izričito navedene u popisu programskih elemenata s pravom pristupa. Programska element s pravom pristupa može biti logički čvor, programska usluga ili primjerak programske usluge.



Slika 7.10 Struktura tablice prava pristupa do lokalnih programskih usluga

Udaljeni pristup s ograničenjima do operacija programskih usluga moguće je ograničavati na tri razine. Pristup je moguće ograničiti na razini logičkih čvorova, programskih usluga i primjeraka programskih usluga. Prava pristupa nekog programskog elementa prenose se na sve programske elemente niže razine. Ako je pristup ograničen na razini logičkog čvora, onda sve programske usluge smještene na logičkom čvoru koji ostvaruje prava pristupa također ostvaruju prava pristupa do operacije pozvane programske usluge. Isto vrijedi i za sve njihove primjerke usluga. Ako je pristup ograničen na razini programskih usluga, onda samo one usluge koje su izričito navedene u popisu programskih elemenata s pravom pristupa ostvaruju prava pristupa do operacije pozvane usluge. Istovremeno, svi primjerici programske usluge koja ostvaruje prava pristupa također ostvaruju prava pristupa do navedene operacije. Ako su prava pristupa ograničena na razini primjeraka usluga, onda samo oni primjerici programskih usluga koji su izričito navedeni u popisu programskih elemenata s pravom pristupa ostvaruju mogućnost pristupa do tražene operacije pozvane programske usluge.

Opisani model ograničavanja prava pristupa do programskih usluga provodi se pravilima definiranim u tablici prava pristupa. Tablica prava pristupa do programskih usluga smještenih na lokalnom logičkom čvoru ima hijerarhijsku strukturu, a zapisana je u obliku *XML* dokumenta koji omogućava provedbu učinkovitih postupaka pretraživanja podataka. Struktura tablice prava pristupa prikazana je slikom 7.10. Cjelokupna tablica podijeljena je u nekoliko cjelina. Svakom cjelinom definirana su prava pristupa do programskih usluga smještenih na jednom logičkom čvoru. Ukupan broj cjelina odgovara broju logičkih čvorova koji su postavljeni na fizičkom čvoru na kojem je postavljena programska usluga prividne mreže.

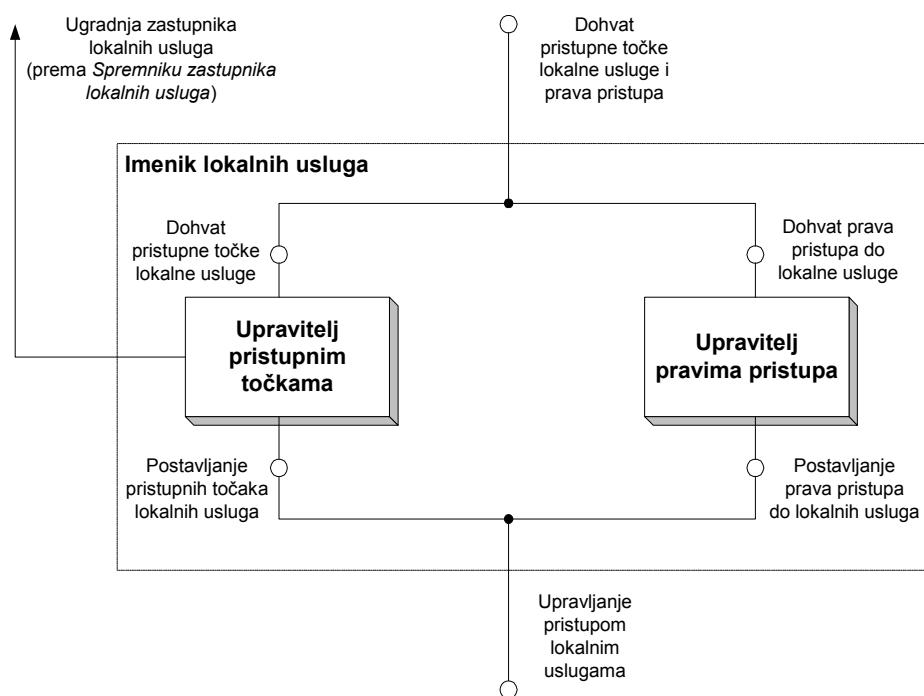
Pretraživanje tablice prava pristupa obavlja se pomoću tri ključa pretraživanja. Prvi ključ pretraživanja je ime lokalnog logičkog čvora na kojem je smještena pozvana programska usluga. Drugi ključ pretraživanja predstavlja logičko ime pozvane programske usluge, dok je treći ključ pretraživanja predstavljen imenom pozvane operacije nad programskom uslugom. Nakon te tri razine pretraživanja dolazi se do podataka o definiranim pravima pristupa. Za svaku operaciju definiran je način ostvarivanja prava pristupa koji može biti lokalni pristup, udaljeni pristup ili udaljeni pristup s ograničenjima. Ako je način ostvarivanja prava pristupa do operacije programske usluge postavljen na lokalni ili udaljeni pristup, time su u potpunosti definirana pravila pristupa. Ako je način ostvarivanja prava pristupa postavljen na udaljeni pristup s ograničenjima, slijedi popis pozivajućih programskih elemenata koji ostvaruju potrebna prava pristupa. Svaki pozivajući programski element određen je s jednom, dvije ili tri razine ograničenja, ovisno o tome da li se pristup ograničava na razini logičkog čvora, programske usluge ili primjerkom programske usluge.

7.2.3 Arhitektura imenika lokalnih usluga

Imenik lokalnih usluga izlaže prema ostalim komponentama programske usluge prividne mreže dva programska sučelja. Sučelje za upravljanje pristupom lokalnim uslugama koristi se tijekom postavljanja nove programske usluge na lokalni logički čvor. Putem tog sučelja, u *Imenik lokalnih usluga* se upisuju, brišu i izmjenjuju podaci o adresama pristupnih točaka novopostavljenih programskih usluga te podaci o pravima pristupa do tih usluga. Te podatke upisuje, briše ili izmjenjuje sustav za postavljanje usluga i sustav za prevođenje i izvođenje raspodijeljenih programa prividne raspodijeljene računalne okoline. Isto sučelje koristi i *Upravitelj članstvom prividne mreže* tijekom ulaska novog logičkog čvora u mrežu i njegova postavljanja na lokalni fizički čvor te tijekom napuštanja prividne mreže od strane logičkog čvora smještenog na lokalnom fizičkom čvoru. Drugo programsko sučelje služi za dohvrat podataka o pristupnim točkama lokalnih usluga i njihovim pravima pristupa. Te

podatke koristi *Usmjernik poruka* u postupku isporuke poruka lokalnim programskim uslugama.

Imenik lokalnih usluga za ostvarivanje svojih funkcionalnosti koristi programski izlaz prema *Spremniku zastupnika lokalnih usluga*. Tijekom postavljanja nove programske usluge na lokalni logički čvor, za tu je uslugu potrebno ugraditi zastupnik putem kojeg joj *Usmjernik poruka* isporučuje primljene poruke. Tijekom uklanjanja programske usluge s lokalnog logičkog čvora, uklanja se i odgovarajući zastupnik. *Imenik lokalnih usluga* po primitku zahtjeva za upisom podataka o pristupnoj točki i pravima pristupa za novu programsku uslugu na svojem sučelju za upravljanje pristupom lokalnim uslugama šalje poruku *Spremniku zastupnika lokalnih usluga* kako bi se ugradio zastupnik za tu uslugu. Prilikom primanja zahtjeva za brisanje podataka o pristupnoj točki i pravima pristupa do programske usluge, *Imenik lokalnih usluga* šalje poruku *Spremniku zastupnika lokalnih usluga* kako bi se uklonio zastupnik odgovarajuće lokalne usluge.



Slika 7.11 Arhitektura *Imenika lokalnih usluga*

Arhitektura *Imenika lokalnih usluga* prikazana je slikom 7.11. Dvije osnovne funkcionalnosti *Imenika lokalnih uluga* podijeljene su između dvije podkomponente. *Upravitelj pristupnim točkama* je programska podkomponenta *Imenika lokalnih usluga* koja je zadužena za upravljanje podacima o pristupnim točkama do programskih usluga postavljenih na lokalnim logičkim čvorovima. Unutrašnju strukturu podataka *Upravitelja*

pristupnim točkama čini tablica pristupnih točaka. Zadatak *Upravitelja pristupnim točkama* je održavanje i pretraživanje te tablice. *Upravitelj pristupnim točkama* izlaže dva programska sučelja koja su povezana na programska sučelja *Imenika lokalnih usluga*. Putem tih dvaju programskih sučelja obavlja se upis, brisanje, izmjena i dohvata podataka o pristupnim točkama lokalnih programskih usluga. Osim upravljanja podacima o pristupnim točkama do lokalnih programskih usluga, *Upravitelj pristupnim točkama* zadužen je za obavještavanje *Spremnika zastupnika lokalnih usluga* o potrebi za ugradnjom, odnosno uklanjanjem zastupnika lokalnih programskih usluga.

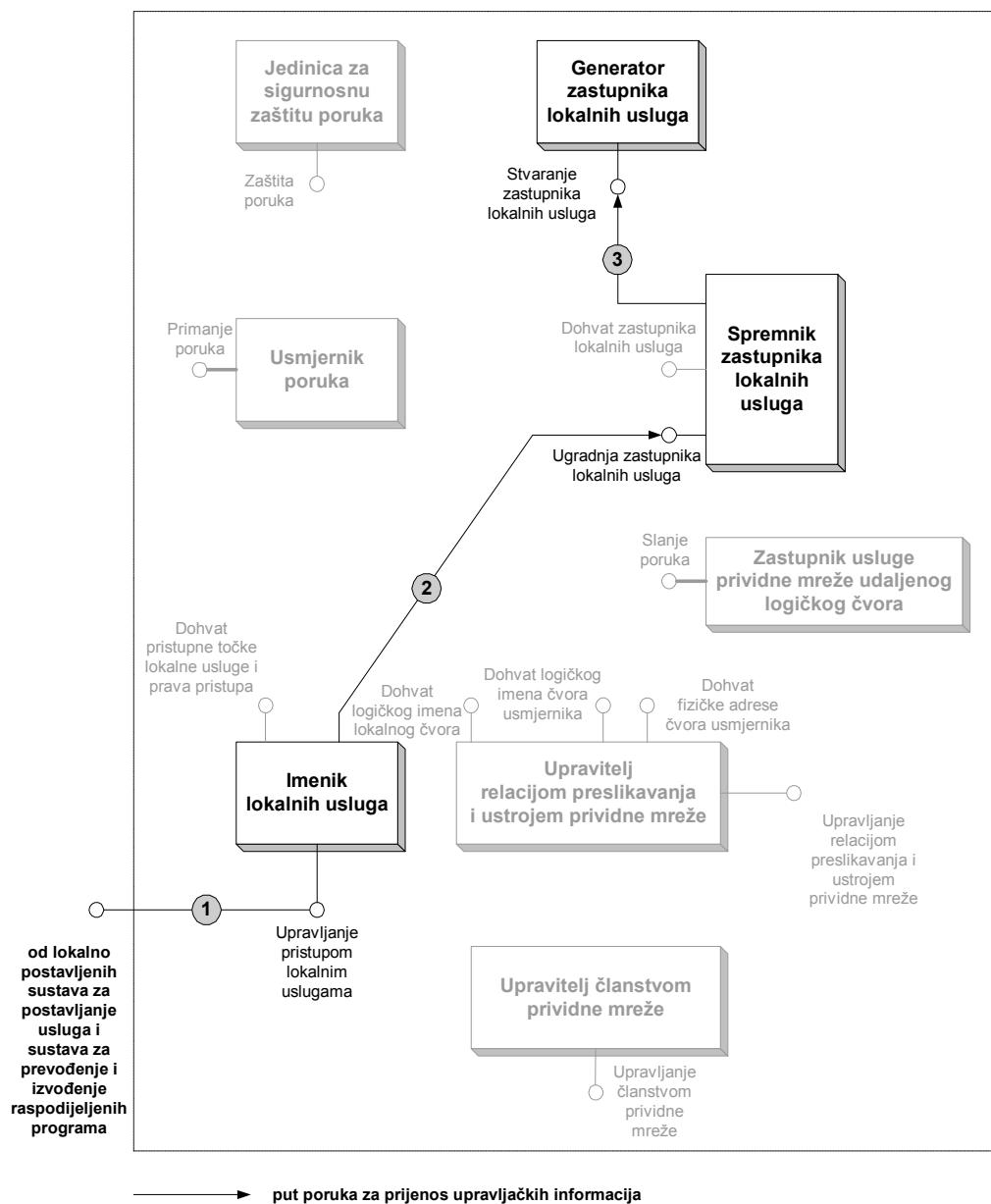
Druga podkomponenta *Imenika lokalnih usluga* je *Upravitelj pravima pristupa*. Uloga *Upravitelja pravima pristupa* je održavanje podataka o pravima pristupa do programskih usluga postavljenih na lokalnim logičkim čvorovima. Unutrašnju strukturu podataka *Upravitelja pravima pristupa* čini tablica prava pristupa. Zadatak *Upravitelja pravima pristupa* je održavanje i pretraživanje te tablice te potvrđivanje ili odbijanje zahtjeva za pristupom pozvanoj programskoj usluzi na osnovi podataka o pozivajućoj programskoj usluzi. Slično *Upravitelju pristupnim točkama*, i *Upravitelj pravima pristupa* izlaže dva programska sučelja. Jedno sučelje koristi se za upis, brisanje i izmjenu prava pristupa, dok se drugo koristi za dohvata upisanih prava pristupa. Oba su sučelja povezana na vanjska sučelja *Imenika lokalnih usluga*.

7.2.4 Prijava lokalnih programskih usluga u sustav prividne mreže

Postavljanje programskih usluga na lokalni logički čvor prividne mreže obavlja sustav za postavljanje usluga. Sustav za prevođenje i izvođenje raspodijeljenih programa također obavlja postupak postavljanja programskih usluga na lokalni logički čvor, budući da raspodijeljeni programi u izvođenju postaju programske usluge. Nakon što je programska usluga postavljena na lokalnom logičkom čvoru, potrebno je osigurati njezinu dostupnost u logičkom komunikacijskom prostoru privine mreže. Kako bi postala dostupna, podatke o adresi njezine fizičke pristupne točke na lokalnom čvoru fizičke mreže i pravima pristupa potrebno je upisati u *Imenik lokalnih usluga* te u *Spremnik zastupnika lokalnih usluga* ugraditi odgovarajući zastupnik usluge. Dostupnost programske usluge u komunikacijskom prostoru prividne mreže osigurava se postupkom prijave programske usluge u sustav prividne mreže na lokalnom logičkom čvoru.

Postupak prijave programske usluge u sustav prividne mreže na lokalnom logičkom čvoru obavlja sustav za postavljanje usluga, odnosno sustav za prevođenje i izvođenje raspodijeljenih programa. Postupak je prikazan slikom 7.12, a sastoji se od tri koraka. Nakon završenog postupka postavljanja programske usluge na lokalni logički čvor, sustav za

postavljanje usluga ili sustav za prevođenje i izvođenje raspodijeljenih programa pokreće postupak prijave usluge ili raspodijeljenog programa u sustav prividne mreže.



Slika 7.12 Postupak prijave lokalno postavljene programske usluge u sustav prividne mreže na lokalnom logičkom čvoru

U prvom koraku pristupa se sučelju za upravljanje pristupom programskim uslugama *Imeniku lokalnih usluga*. Pri pristupanju tom sučelju, *Imeniku lokalnih usluga* se predaju podaci o logičkom imenu novopostavljene programske usluge, adresi njezine pristupne točke, pravima pristupa te dokument s opisom programske usluge u obliku *WSDL* dokumenta. Podatke o logičkom imenu usluge i adresi pristupne točke *Imenik lokalnih*

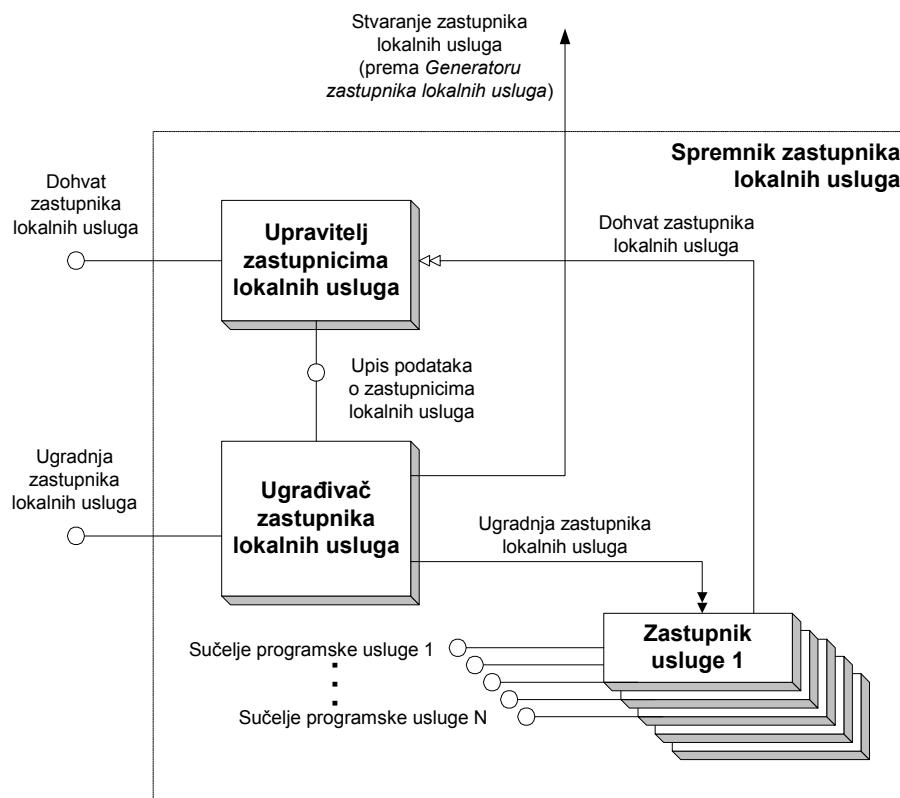
usluga upisuje u tablicu pristupnih točaka. Podaci o pravima pristupa upisuju se u tablicu prava pristupa. Dokument s opisom programske usluge prosljeđuje se *Spremniku zastupnika lokalnih usluga* kako bi on na osnovi podataka sadržanih u tom dokumentu izgradio odgovarajući zastupnik za zadalu uslugu. Prosljeđivanje dokumenta s opisom programske usluge *Spremniku zastupnika lokalnih usluga* obavlja se u drugom koraku postupka prijave. Po primljenom dokumentu od *Imenika lokalnih usluga*, *Spremnik zastupnika lokalnih usluga* u trećem koraku pristupa do *Generatora zastupnika lokalnih usluga*, prosljeđujući mu primljeni dokument. *Generator zastupnika lokalnih usluga* na osnovi dokumenta s opisom programske usluge stvara programski kôd zastupnika zadane usluge, prevodi ga te ga u izvodivom obliku vraća *Spremniku zastupnika lokalnih usluga*. Dobiveni izvodivi kôd zastupnika lokalne usluge *Spremnik zastupnika lokalnih usluga* ugrađuje u svoje podatkovne strukture kako bi bili dostupni *Uzmjerniku poruka* u postupku isporuke poruka lokalnim programskim uslugama.

Nakon uklanjanja programske usluge s lokalnog logičkog čvora, njezine podatke potrebno je ukloniti iz sustava prividne mreže. Postupak uklanjanja podataka iz sustava sličan je postupku prijave usluge u sustav. U prvom koraku postupka, sustav za postavljanje usluga, odnosno sustav za prevođenje i izvođenje raspoloženih programa pristupa *Imeniku lokalnih usluga*, obavještavajući ga o uklonjenoj programskoj usluzi. *Imenik lokalnih usluga* uklanja podatke o uklonjenoj programskoj usluzi iz tablice pristupnih točaka i tablice prava pristupa. Nakon toga, u drugom koraku postupka, *Imenik lokalnih usluga* pristupa *Spremniku zastupnika lokalnih usluga* kako bi se zastupnik programske usluge uklonio iz sustava.

7.3 Spremnik zastupnika lokalnih usluga

Spremnik zastupnika lokalnih usluga je programska komponenta usluge prividne mreže koja upravlja zastupnicima programskih usluga (engl. *service proxy*) postavljenih na lokalnim logičkim čvorovima. Arhitektura *Spremnika zastupnika lokalnih usluga* prikazana je slikom 7.13. Prema ostalim komponentama usluge prividne mreže, *Spremnik zastupnika lokalnih usluga* izlaže dva programska sučelja. Jedno programsko sučelje služi za ugradnju zastupnika lokalnih usluga. Putem tog sučelja postavljaju se zahtjevi za ugradnjom zastupnika lokalnih usluga u *Spremniku zastupnika lokalnih usluga*, odnosno za njihovo uklanjanje iz *Spremnika zastupnika lokalnih usluga*. Sučelje za ugradnju zastupnika lokalnih usluga koristi *Imenik lokalnih usluga* u postupku prijave programskih usluga u sustav prividne mreže lokalnog logičkog čvora, odnosno u postupku njihova uklanjanja iz sustava. Drugo programsko sučelje koje izlaže *Spremnik zastupnika lokalnih usluga* koristi se za

dohvat ugrađenih zastupnika. To sučelje koristi *Usmjernik poruka* tijekom isporuke poruka lokalnim programskim uslugama. Povezanost *Spremnika zastupnika lokalnih usluga* s ostalim komponentama programske usluge prividne mreže zaokružuje programski izlaz prema *Generatoru zastupnika lokalnih usluga*. Putem tog programskog izlaza *Spremnik zastupnika lokalnih usluga* poziva *Generator zastupnika lokalnih usluga* u cilju prijavljivanja izvodivog programskog kôda zastupnika zadane programske usluge.



Slika 7.13 Arhitektura *Spremnika zastupnika lokalnih usluga*

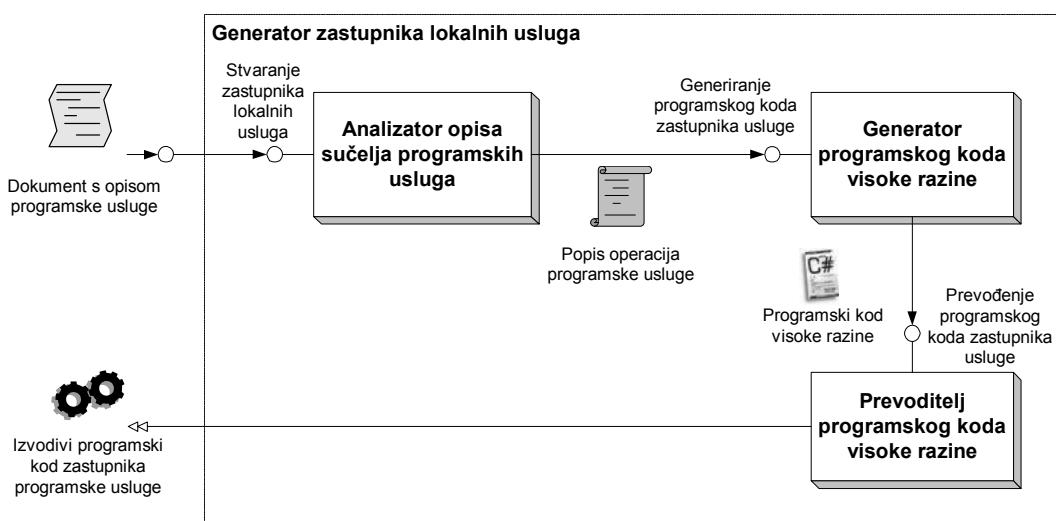
Spremnik zastupnika lokalnih usluga podijeljen je na dvije podkomponente. *Ugrađivač zastupnika lokalnih usluga* je podkomponenta zadužena za provedbu postupka ugradnje i uklanjanja zastupnika lokalnih usluga u i iz sustava prividne mreže lokalnog logičkog čvora. Po zaprimljenom zahtjevu na sučelju za ugradnju zastupnika lokalnih usluga, *Ugrađivač zastupnika lokalnih usluga* uspostavlja vezu s *Generatorom zastupnika lokalnih usluga* i od njega prijavlja odgovarajući zastupnik u obliku izvodivog programskog kôda. Od prijavljenog izvodivog programskog kôda oblikuje datoteku, dodjeljuje joj ime te je ugrađuje u datotečni podsustav operacijskog sustava lokalnog fizičkog čvora. Nakon ugradnje zastupnika u datotečni podsustav lokalnog operacijskog sustava, *Ugrađivač zastupnika lokalnih usluga* podatke o ugrađenoj datoteci upisuje u *Upravitelj zastupnicima*.

lokalnih usluga. Upravitelj zastupnicima lokalnih usluga je druga podkomponenta *Spremnika zastupnika lokalnih usluga*. Njegova je uloga održavanje podataka o ugrađenim zastupnicima te isporuka izvodivih programskih kôdova zastupnika lokalnih usluga *Usmjerniku poruka*. *Upravitelj zastupnicima lokalnih usluga* održava jednostavnu tablicu u kojoj za svaku lokalnu programsku uslugu postoje podaci o njezinom logičkom imenu i imenu datoteke koja sadrži izvodivi programski kôd njezina zastupnika. U postupku isporuke poruke lokalnoj programskoj usluzi, *Usmjernik poruka* od *Spremnika zastupnika lokalnih usluga* dohvaća zastupnik zadane usluge. Zahtjevi *Usmjernika poruka* pristižu na sučelje za dohvati zastupnika lokalnih usluga *Upravitelja zastupnicima lokalnih usluga*. Na osnovi logičkog imena programske usluge koje je sadržano u zahtjevu, *Upravitelj zastupnicima lokalnih usluga* u svojoj tablici pronalazi ime datoteke s izvodivim programskim kôdom zastupnika, dohvaća kôd iz datoteke i šalje ga *Usmjerniku poruka*.

7.4 Generator zastupnika lokalnih usluga

Generator zastupnika lokalnih usluga je programska komponenta usluge prividne mreže koja je zadužena za stvaranje zastupnika programskih usluga postavljenih na lokalnim logičkim čvorovima. Zastupnici lokalnih usluga stvaraju se na osnovi dokumenata s opisom programske usluge, a izgrađeni su u obliku izvodivog programskog kôda. Arhitektura *Generatora zastupnika lokalnih usluga* i postupak stvaranja zastupnika prikazani su slikom 7.14. *Generator zastupnika lokalnih usluga* je samostalna programska komponenta koja sve funkcionalnosti obavlja bez potrebe za suradnjom s drugim komponentama ili programskim uslugama. Iz tog razloga *Generator zastupnika lokalnih usluga* nema definiranih programskih izlaza prema ostalim dijelovima sustava. Ostale komponente usluge prividne mreže suradnju s *Generatorem zastupnika lokalnih usluga* ostvaruju putem njegovog programskog sučelja za stvaranje zastupnika lokalnih usluga. Ovo programsko sučelje koristi *Spremnik zastupnika lokalnih usluga* tijekom ugradnje zastupnika lokalnih usluga u svoje spremničke strukture podataka. Ulazni parametar poziva *Generatora zastupnika lokalnih usluga* na sučelju za stvaranje zastupnika lokalnih usluga je dokument s opisom programske usluge za koju je potrebno izgraditi zastupnik. Dokument s opisom programske usluge napisan je u *WSDL* jeziku. Rezultat poziva *Generatora zastupnika lokalnih usluga* je zastupnik zahtjevane programske usluge u obliku izvodivog programskog kôda. Izgrađeni zastupnik programske usluge spremjan je za ugradnju u *Spremnik zastupnika lokalnih usluga* i za kasnije korištenje od strane *Usmjernika poruka*.

Stvaranje zastupnika lokalne usluge je slijedni postupak koji se sastoji od tri koraka. Svaki korak postupka obavlja izdvojena podkomponenta *Generatora zastupnika lokalnih usluga*. U prvom koraku postupka obavlja se analiza dokumenta s opisom programske usluge. Analizu tog dokumenta obavlja podkomponenta *Generatorka zastupnika lokalnih usluga* pod nazivom *Analizator opisa sučelja programskih usluga*. Analizom dokumenta dohvaćaju se operacije koje programska usluga izlaže na korištenje ostalim programskim uslugama. *Analizator opisa sučelja programskih usluga* za slijedeći korak postupka izgradnje zastupnika priprema dokument s popisom pronađenih operacija.



Slika 7.14 Arhitektura *Generatorka zastupnika lokalnih usluga* s prikazom postupka izgradnje zastupnika

U drugom koraku pristupa se generiranju programskog kôda zastupnika lokalne usluge u programskom jeziku visoke razine. Na osnovi dokumenta s popisom operacija programske usluge generiraju se naredbe programskog jezika visoke razine. Generiranje programskog kôda visoke razine obavlja *Generator programskog kôda visoke razine*. U postupku generiranja programskog kôda visoke razine, *Generator programskog kôda visoke razine* koristi unaprijed pripremljene predloške programskog kôda zastupnika. U pripremljenim predlošcima su ostavljena prazna mjesta koja se odnose na imena operacija programske usluge. Zadatak *Generatorka programskog kôda visoke razine* je popunjavanje tih praznima podacima iz popisa koje prima na svom ulaznom sučelju.

Treći korak postupka izgradnje zastupnika lokalne usluge koristi se za prevođenje naredaba programskog jezika visoke razine u naredbe izvodivog programa. Za provedbu tog koraka zadužena je treća podkomponenta *Generatorka lokalnih usluga* pod nazivom *Prevoditelj programskog kôda visoke razine*. Nakon završenog postupka

prevodenja, *Prevoditelj programskog kôda visoke razine* na izlazu daje izvodivi programski kôd zastupnika zadane programske usluge. Zastupnik lokalne usluge u izvodivom obliku vraća se u *Spremnik zastupnika lokalnih usluga* kao povratna vrijednost poziva upućenog *Generatoru zastupnika lokalnih usluga*.

7.5 Upravitelj relacijom preslikavanja i ustrojem prividne mreže

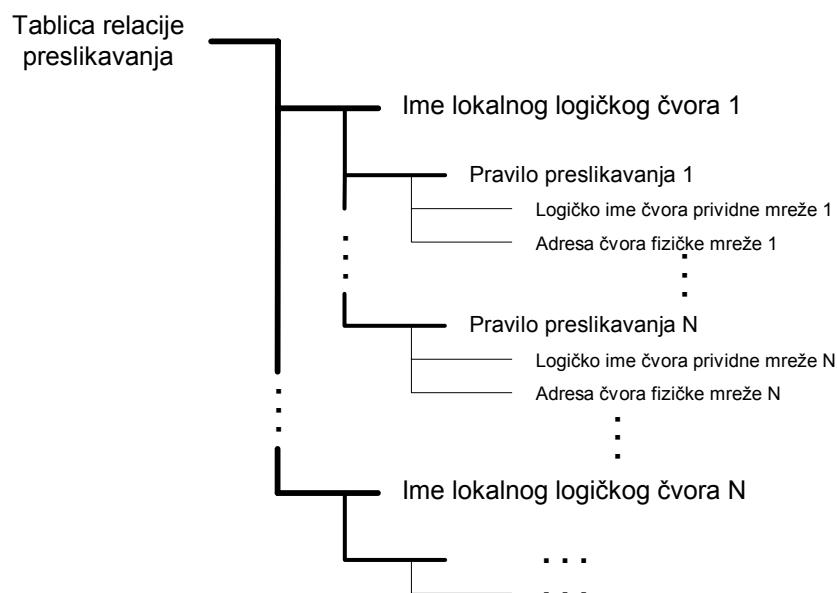
Upravitelj relacijom preslikavanja i ustrojem prividne mreže je komponenta programske usluge prividne mreže zadužena za upravljanje podacima o relaciji preslikavanja prividne na fizičku mrežu te upravljanje podacima za usmjeravanje poruka među logičkim čvorovima. Za upravljanje spomenutim podacima, *Upravitelj relacijom preslikavanja i ustrojem prividne mreže* koristi dvije tablice. Podaci o relaciji preslikavanja prividne na fizičku mrežu sadržani su u tablici relacije preslikavanja. Podaci o usmjeravanju poruka među logičkim čvorovima sadržani su u tablici usmjeravanja. U sustavu prividne mreže svaki logički čvor ima vlastitu, logički odvojenu tablicu relacije preslikavanja i tablicu usmjeravanja. Podaci u tim dvjema tablicama mogu biti isti za sve logičke čvorove, a mogu se i međusobno razlikovati. Sadržaj tablice relacije preslikavanja i tablice usmjeravanja ovisi o algoritmima za razmjenu podataka o relaciji preslikavanja, ustroju prividne mreže te usmjeravanju poruka među logičkim čvorovima.

7.5.1 Tablica relacije preslikavanja

Relacija preslikavanja prividne na fizičku mrežu određuje način preslikavanja imena logičkih čvorova u adrese fizičkih čvorova. Svojstvo relacije preslikavanja je preslikavanje više na jedan (engl. *many to one*). Time je omogućeno preslikavanje više logičkih imena u istu fizičku adresu, odnosno postavljanje proizvoljno mnogo logičkih čvorova prividne mreže na jedan čvor fizičke mreže. Podaci o relaciji preslikavanja prividne na fizičku mrežu sadržani su u tablici relacije preslikavanja. Tablica relacije preslikavanja je hijerarhijski organizirana struktura podataka. U radnom spremniku računala spremljena je u obliku *XML* dokumenta. Podijeljena je u nekoliko osnovnih cjelina, pri čemu svaka cjelina pripada jednom logičkom čvoru postavljenom na lokalnom fizičkom čvoru. Svaka cjelina tablice relacije preslikavanja sadrži podatke o relaciji preslikavanja koje koristi odgovarajući lokalni logički čvor.

Struktura tablice relacije preslikavanja prikazana je slikom 7.15. Svaka cjelina tablice sastoji se od određenog broja pravila preslikavanja. U najjednostavnijem slučaju, broj pravila preslikavanja jednak je ukupnom broju svih logičkih čvorova prividne mreže. U općem

slučaju, dovoljan broj pravila preslikavanja jednak je broju logičkih čvorova s kojima promatrani logički čvor ostvaruje izravne logičke komunikacijske veze. Svako pravilo preslikavanja određeno je uređenim parom podataka koji se sastoji od logičkog imena čvora prividne mreže i adrese čvora fizičke mreže na kojem je taj logički čvor postavljen. U trenutku kada *Usmjernik poruka* treba primljenu poruku isporučiti udaljenom logičkom čvoru, logičko ime tog čvora se zamjenjuje njegovom fizičkom adresom pronađenom u tablici relacije preslikavanja.



Slika 7.15 Struktura tablice relacije preslikavanja

7.5.2 Tablica usmjeravanja

Pravila usmjeravanja poruka u sustavu prividne mreže moguće je definirati na tri načina. Usmjeravanje poruke s jednog logičkog čvora usmjernika na drugi logički čvor usmjernik može biti ovisno o izvorištu i odredištu poruke, ovisno samo o odredištu poruke, ili neovisno o izvorištu i odredištu poruke. Najopćenitiji slučaj pravila usmjeravanja je pravilo usmjeravanja ovisno o izvorištu i odredištu poruke. Uporabom ove vrste pravila usmjeravanja, odluka o odabiru čvora usmjernika putem kojeg se nastavlja put poruke od izvorišnog do odredišnog logičkog čvora donosi se na osnovi logičkih imena izvorišnog i odredišnog logičkog čvora. Za svaku kombinaciju logičkog imena izvorišnog i odredišnog logičkog čvora definira se po jedno pravilo usmjeravanja. Pravilima usmjeravanja ovisnim o izvorištu i odredištu poruke moguće je oblikovati najprilagodljiviji ustroj prividne mreže nad fizičkom mrežom. Nedostatak primjene ovih pravila je porast količine prostora u radnom

spremniku računala koja je potrebna za njihovo spremanje. Količina potrebnog spremničkog prostora raste s kvadratom broja čvorova prividne mreže.

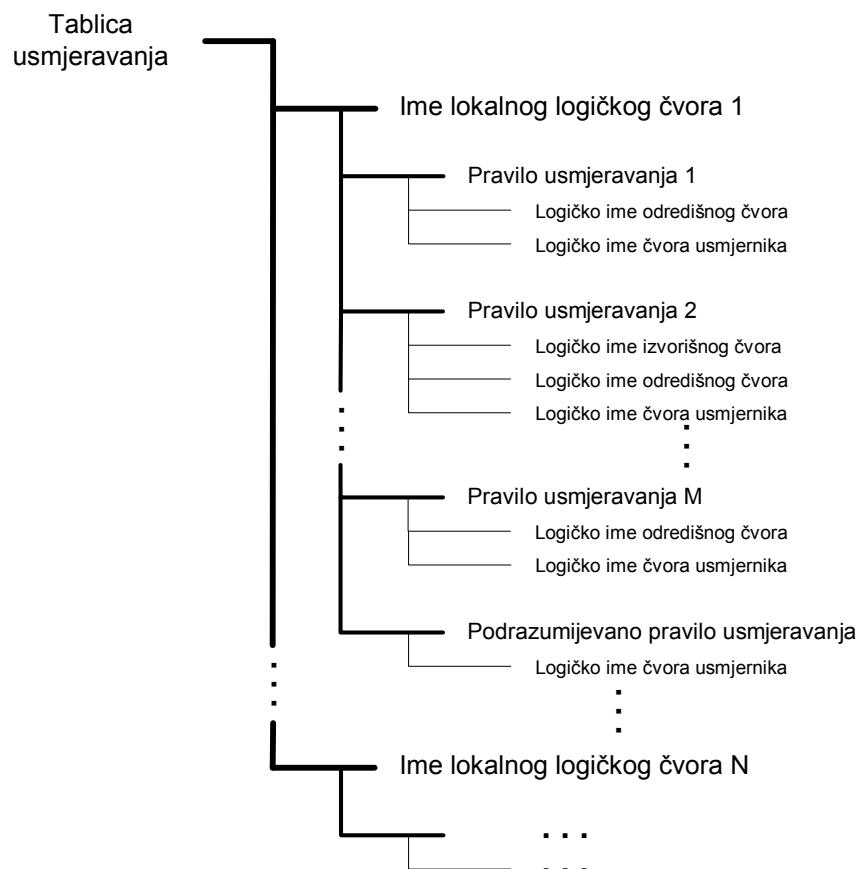
Učinkovitije gospodarenje potrošnjom prostora u radnom spremniku računala postiže se primjenom pravila usmjeravanja ovisnih samo o odredištu poruke. Primjenom ove vrste pravila usmjeravanja, odluka o odabiru čvora usmjernika putem kojeg se nastavlja put poruke od izvorišnog do odredišnog logičkog čvora donosi se na osnovi logičkog imena odredišnog čvora. Budući da je u ovom slučaju po svakom logičkom čvoru dovoljno uvesti po jedno pravilo usmjeravanja, potrošnja prostora u radnom spremniku računala raste po linearnom zakonu u odnosu na povećanje broja čvorova prividne mreže.

Najmanja potrošnja prostora u radnom spremniku računala postiže se uporabom pravila usmjeravanja neovisnih o izvorištu i odredištu poruke. Takva se pravila nazivaju podrazumijevanim pravilima usmjeravanja. Uporabom podrazumijevanih pravila usmjeravanja, primljene se poruke uvijek prosljeđuju unaprijed određenom logičkom čvoru usmjerniku. Primjena podrazumijevanih pravila usmjeravanja dopušta vrlo ograničene mogućnosti prilagodbe ustroja prividne mreže. Primjerice, ustroj prividne mreže s logičkim čvorovima organiziranim u prstenastu strukturu i samo jednim dopuštenim smjerom obilaženja prstena pogodan je za oblikovanje pomoću podrazumijevanih pravila usmjeravanja. Podrazumijevana pravila usmjeravanja najčešće primjenu pronalaze kao nadopuna pravilima usmjeravanja ovisnima o odredištu poruke.

Podaci o pravilima usmjeravanja spremljeni su u tablici usmjeravanja. Tablica usmjeravanja je hijerarhijska struktura podataka, a u računalu je predstavljena u obliku *XML* dokumenta. Podijeljena je na onoliko osnovnih cjelina koliko logičkih čvorova je postavljeno na lokalnom čvoru fizičke mreže. Jednom cjelinom tablice usmjeravanja definirana su pravila usmjeravanja poruka koja tijekom usmjeravanja koristi jedan od lokalnih logičkih čvorova. Struktura tablice usmjeravanja prikazana je slikom 7.16.

Bilo koja cjelina tablice usmjeravanja sastoji se od određenog broja pravila usmjeravanja. Broj pravila usmjeravanja ovisi o broju logičkih čvorova u sustavu prividne mreže, o ustroju prividne mreže te o vrsti korištenih pravila. Pravilo usmjeravanja ovisno samo o odredištu poruke definirano je uređenim parom podataka. Prvi podatak određuje ime odredišnog logičkog čvora i predstavlja ključ za pronalaženje pravila. Drugim podatkom određeno je ime logičkog čvora usmjernika. Primjer pravila usmjeravanja ovisnog samo o odredištu poruke je prvo pravilo usmjeravanja u primjeru na slici 7.16. Pravilo usmjeravanja ovisno o izvorištu i odredištu poruke definirano je uređenom trojkom podataka. Prvi podatak određuje ime izvorišnog, a drugi ime odredišnog logičkog čvora. Oba podatka predstavljaju

ključ za pronalaženje pravila. Trećim članom uređene trojke određeno je ime logičkog čvora usmjernika. Primjer pravila usmjeravanja ovisnog o izvorištu i odredištu poruke je drugo pravilo usmjeravanja u primjeru na slici 7.16. Podrazumijevano pravilo usmjeravanja definirano je samo jednom vrijednošću koja predstavlja ime logičkog čvora usmjernika. Primjer podrazumijevanog pravila usmjeravanja je posljednje pravilo usmjeravanja u primjeru na slici 7.16.

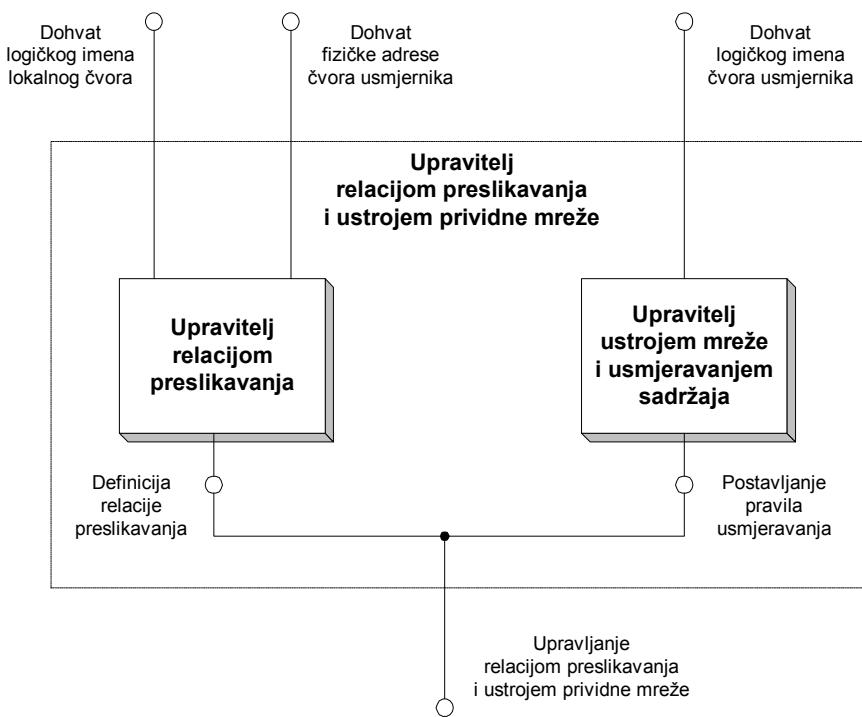


Slika 7.16 Struktura tablice usmjeravanja

Traženje odgovarajućeg pravila usmjeravanja u tablici usmjeravanja započinje pokušajem pronalaska odgovarajućeg pravila usmjeravanja ovisnog o izvorištu i odredišu poruke. Ako takvo pravilo nije pronađeno, pokušava se pronaći odgovarajuće pravilo ovisno samo o odredištu poruke. Ako oba pokušaja pronalaska pravila usmjeravanja završe neuspješno, koristi se podrazumijevano pravilo usmjeravanja. Po jednom logičkom čvoru dozvoljeno je definirati najviše jedno podrazumijevano pravilo usmjeravanja.

7.5.3 Arhitektura upravitelja relacijom preslikavanja i ustrojem prividne mreže

Upravitelj relacijom preslikavanja i ustrojem prividne mreže koristi se u sprezi s *Usmjernikom poruka* u postupku isporuke poruka udaljenim logičkim čvorovima. Podaci koje *Upravitelj relacijom preslikavanja i ustrojem prividne mreže* osigurava *Usmjerniku poruka* su informacije o imenu lokalnog logičkog čvora, imenu logičkog čvora usmjernika i adresi fizičkog čvora usmjernika. Ime lokalnog logičkog čvora potrebno je *Usmjerniku poruka* u postupku usporedbe imena odredišnog logičkog čvora s imenom lokalnog logičkog čvora. Logičko ime čvora usmjernika potrebno je *Usmjerniku poruka* tijekom prosljeđivanja poruka udaljenim logičkim čvorovima. Adresa fizičkog čvora usmjernika potrebna je za fizički prijenos podataka računalnom mrežom od lokalnog fizičkog čvora do fizičkog čvora na kojem je smješten logički čvor usmjernik.



Slika 7.17 Arhitektura *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*

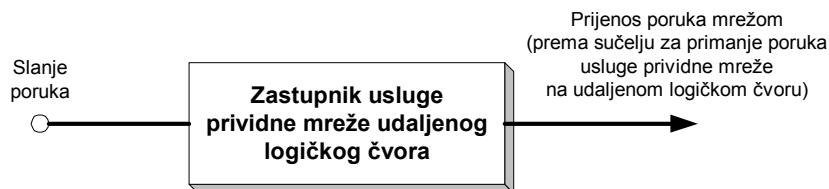
Arhitektura *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* prikazana je slikom 7.17. Suradnja s ostalim komponentama programske usluge prividne mreže, kao i ostalim programskim uslugama ostvarena je putem četiri programska sučelja. Dohvat imena lokalnog logičkog čvora *Usmjernik poruka* dohvatača pristupanjem sučelju za dohvat logičkog imena lokalnog čvora. Pribavljanje podataka o logičkom imenu i fizičkoj adresi čvora usmjernika omogućeno je pristupanjem sučeljima za dohvat logičkog imena čvora

usmjernika, odnosno sučelju za dohvat fizičke adrese čvora usmjernika. Pristupanjem sučelju za upravljanje relacijom preslikavanja i ustrojem prividne mreže, *Upravitelju relacijom preslikavanja i ustrojem prividne mreže* se dostavljaju podaci o dopunama i izmjenama tablice relacije preslikavanja i tablice usmjeravanja.

Upravitelj relacijom preslikavanja i ustrojem prividne mreže je složena programska komponenta čija je unutrašnja građa podijeljena na dvije podkomponente. Svaka od dviju podkomponenti zadužena je za obavljanje jedne od dviju osnovnih funkcionalnosti. *Upravitelj relacijom preslikavanja* zadužen je za održavanje, ažuriranje i pretraživanje tablice relacije preslikavanja. *Upravitelj ustrojem mreže i usmjeravanjem sadržaja* zadužen je za održavanje, ažuriranje i pretraživanje tablice usmjeravanja. Svaka od dviju podkomponenti *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* izlaže programska sučelja za dohvat i ažuriranje podataka u tablici koju održava.

7.6 Zastupnik usluge prividne mreže udaljenog logičkog čvora

Zastupnik usluge prividne mreže udaljenog logičkog čvora služi za prijenos poruka računalnom mrežom između dvaju logičkih čvorova. Logička komunikacijska veza između logičkog čvora pošiljatelja poruke i logičkog čvora primatelja poruke uspostavlja se pozivom programske usluge prividne mreže s logičkog čvora pošiljatelja na logičkom čvoru primatelju. U skladu s općim modelom pozivanja programskih usluga prikazanim slikom 7.6, za poziv programske usluge potrebno je izgraditi zastupnik usluge na strani pozivatelja koji poruke sa zahtjevima i odgovorima prenosi između pozivajuće i pozvane usluge. Budući da upućivanje poruke između dvaju logičkih čvorova odgovara pozivu programske usluge prividne mreže s lokalnog na udaljeni logički čvor, za slanje poruke udaljenom logičkom čvoru potrebno je izgraditi zastupnik programske usluge prividne mreže. Uloga zastupnika usluge prividne mreže u postupku isporuke poruke udaljenom logičkom čvoru jednak je ulogama zastupnika lokalnih usluga ugrađenih u *Spremnik zastupnika lokalnih usluga* u postupku isporuke poruka lokalnim programskim uslugama.



Slika 7.18 Arhitektura Zastupnika usluge prividne mreže udaljenog logičkog čvora

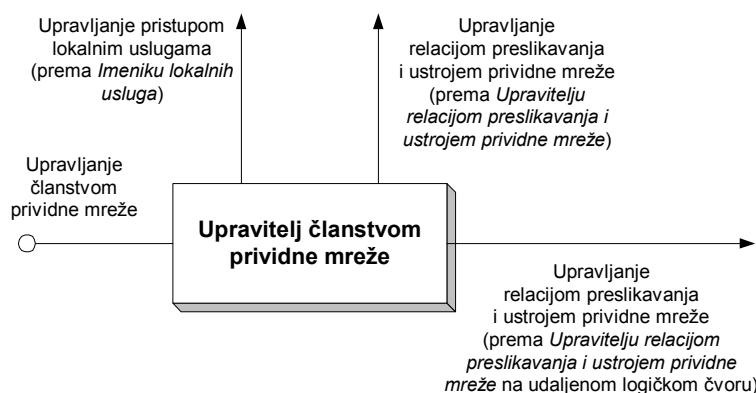
Slikom 7.18 prikazana je arhitektura *Zastupnika usluge prividne mreže udaljenog logičkog čvora*. To je jednostavna programska komponenta čiji je zadatak prijenos poruka računalnom mrežom s lokalnog na udaljeni logički čvor. Putem sučelja za slanje poruka koje izlaže, *Zastupnik usluge prividne mreže udaljenog logičkog čvora* zaprima poruke od *Usmjernika poruka* na lokalnom logičkom čvoru te ih upućuje na sučelje za primanje poruke *Usmjernika poruka* na udaljenom logičkom čvoru. Parametri koje *Usmjernik poruka* proslijeduje *Zastupniku usluge prividne mreže udaljenog logičkog čvora* tijekom pristupanja sučelju za slanje poruka je poruka koju je potrebno prenijeti do udaljenog logičkog čvora te adresa fizičkog čvora na kojem se nalazi taj logički čvor.

7.7 Upravitelj članstvom prividne mreže

Upravitelj članstvom prividne mreže je programska komponenta usluge prividne mreže koja je zadužena za nadzor i provedbu postupka uključivanja novih logičkih čvorova u sustav prividne mreže i postupka napuštanja prividne mreže od strane postojećih logičkih čvorova. Kao što je prikazano slikom 7.19, *Upravitelj članstvom prividne mreže* je jednostavna programska komponenta bez daljnje podjele na podkomponente. Programsko sučelje putem kojeg se ostvaruje veza s *Upraviteljem članstvom prividne mreže* je sučelje za upravljanje članstvom prividne mreže. To je sučelje namijenjeno za korištenje od strane korisnika prividne raspodijeljene računalne okoline putem njegovih pristupnih mehanizama. Pristupanjem sučelju za upravljanje članstvom prividne mreže, korisnik prividne raspodijeljene računalne okoline uključuje i isključuje čvorove fizičke mreže u i iz sustava prividne mreže, postavljajući po njima odgovarajući broj logičkih čvorova. Suradnja *Upravitelja članstvom prividne mreže* s ostalim komponentama usluge prividne mreže ostvarena je putem triju programskih izlaza. Putem jednog programskog izlaza ostvarena je suradnja s *Imenikom lokalnih usluga*, drugim programskim izlazom ostvaruje se suradnja s *Upraviteljem relacijom preslikavanja i ustrojem prividne mreže* na lokalnom logičkom čvoru, dok se treći programski izlaz koristi za ostvarivanje suradnje s *Upraviteljem relacijom preslikavanja i ustrojem prividne mreže* na udaljenim logičkim čvorovima.

Osnovna funkcionalnost koju obavlja *Upravitelj članstvom prividne mreže* je provedba algoritma uključivanja novog logičkog čvora u sustav prividne mreže, odnosno isključivanja postojećeg logičkog čvora iz sustava prividne mreže. Budući da se tijekom uključivanja i isključivanja čvora mijenja ustroj prividne mreže, podatke o novoučlanjenom logičkom čvoru, odnosno logičkom čvoru koji napušta mrežu potrebno je dostaviti ostalim čvorovima prividne mreže. Broj logičkih čvorova s kojima je potrebno razmijeniti informacije o

promjenama u ustroju prividne mreže ovisi o načinu njegova održavanja. Trenutna inačica *Upravitelja članstvom prividne mreže* provodi postupak održavanja potpuno povezane prividne mreže. U potpuno povezanoj prividnoj mreži bilo koji logički čvor ima podatke o postojanju svih ostalih logičkih čvorova i sa svakim je od njih povezan izravnom logičkom komunikacijskom vezom. Za bilo koji čvor prividne mreže može se reći da je u logičkom komunikacijskom prostoru susjedni čvor sa svim ostalim čvorovima prividne mreže. Održavanje potpuno povezane prividne mreže zahtijeva dojavljivanje svih promjena u logičkom ustroju mreže svim logičkim čvorovima.

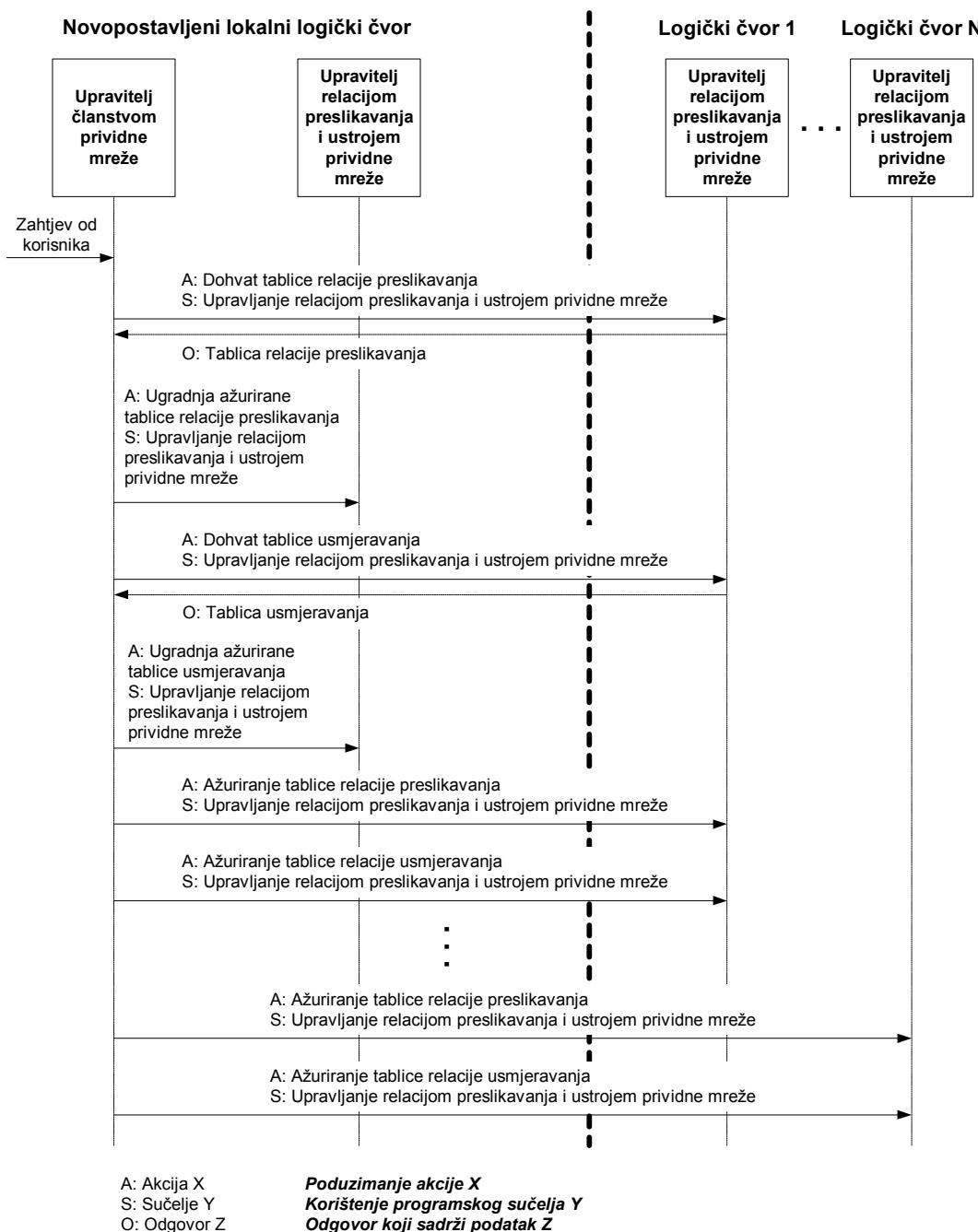


Slika 7.19 Arhitektura *Upravitelja članstvom prividne mreže*

Algoritam održavanja potpuno povezane prividne mreže za slučaj uključivanja novog čvora u mrežu prikazan je slikom 7.20. *Upravitelj članstvom prividne mreže* provodi algoritam za korisničke zahtjeve zaprimljene na sučelju za upravljanje članstvom prividne mreže. Korisnik tijekom postavljanja zahtjeva za uključivanjem novog logičkog čvora u sustav prividne mreže zadaje njegovo logičko ime i fizičku adresu. Pristupni mehanizmi prividne raspodijeljene računalne okoline korisničkom zahtjevu dodaju podatke o logičkom imenu i fizičkoj adresi čvora prividne mreže od kojeg novouključeni čvor treba preuzeti podatke o usmjeravanju poruka i relaciji preslikavanja. Prošireni zahtjev upućuje se na sučelje za upravljanje članstvom prividne mreže *Upravitelja članstvom prividne mreže* na fizičkom čvoru na kojem je potrebno postaviti novi logički čvor.

Na osnovi logičkog imena i fizičke adrese čvora od kojeg je potrebno preuzeti podatke o relaciji preslikavanja, *Upravitelj članstvom prividne mreže* od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* tog logičkog čvora dohvata tablicu relacije preslikavanja. Dobivenu tablicu nadopunjuje vlastitim podacima dodavanjem pravila preslikavanja kojim se njegovo logičko ime preslikava u adresu fizičkog čvora na kojem se nalazi. Ažuriranu tablicu ugrađuje u svoj lokalni *Upravitelj relacijom preslikavanja i*

ustrojem prividne mreže. Nakon toga od *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* logičkog čvora od kojeg preuzima podatke o usmjeravanju poruka dohvata njegovu tablicu usmjeravanja. Pribavljenu tablicu proširuje pravilom usmjeravanja kojim se sve poruke u kojima se lokalni logički čvor nalazi u svojstvu odredišnog čvora prosljeđuju njemu samome. Nakon proširivanja, tablicu usmjeravanja ugrađuje u lokalni *Upravitelj relacijom preslikavanja i ustrojem prividne mreže*.



Slika 7.20 Algoritam uključivanja novog logičkog čvora u sustav prividne mreže

Pribavljanjem i ugradnjom ažurirane tablice relacije preslikavanja i tablice usmjeravanja u svoj lokalni *Upravitelj relacijom preslikavanja i ustrojem prividne mreže*, novopostavljeni logički čvor postaje logički povezan sa svim ostalim čvorovima prividne mreže. Međutim, da bi i ostali logički čvorovi prepoznali identitet novopostavljenog logičkog čvora te mogli s njim uspostaviti logičku komunikacijsku vezu, njima je potrebno dostaviti podatke o logičkom imenu i fizičkoj adresi tog čvora. Taj postupak *Upravitelj članstvom prividne mreže* provodi u nekoliko slijednih koraka. Budući da se prividna mreža održava potpuno povezanim, broj koraka ovisi o broju postojećih logičkih čvorova u sustavu. U svakom koraku se ažuriraju tablica relacije preslikavanja i tablica usmjeravanja jednog logičkog čvora. Ažuriranje tih tablica se provodi na isti način kao ažuriranje lokalnih tablica. Svakom se logičkom čvoru dojavljuje pravilo preslikavanja novopostavljenog logičkog čvora na njegovu fizičku adresu i pravilo usmjeravanja poruka do novopostavljenog logičkog čvora.

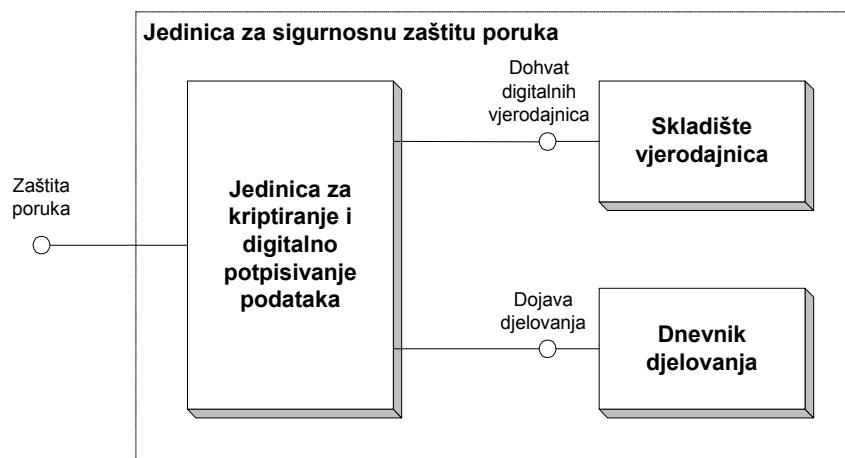
Algoritam održavanja potpuno povezane prividne mreže prilikom isključivanja čvora iz mreže istovjetan je algoritmu koji se provodi prilikom ulaska novog čvora u mrežu. Umjesto dojavljivanja pravila preslikavanja i pravila usmjeravanja kojima postojeći logički čvorovi nadopunjaju vlastite tablice, u ovom se slučaju svim preostalim čvorovima mreže dojavljuju pravila preslikavanja i pravila usmjeravanja koja je potrebno ukloniti iz odgovarajućih tablica.

7.8 Jedinica za sigurnosnu zaštitu poruka

Jedinica za sigurnosnu zaštitu poruka je programska komponenta usluge prividne mreže zadužena za provedbu sigurnosnih zahvata nad *SOAP* porukama. Izlazne *SOAP* poruke koje lokalni logički čvor šalje udaljenim logičkim čvorovima *Jedinica za sigurnosnu zaštitu poruka* proširuje elementima kojima se tim čvorovima jamči njihova tajnost, vjerodostojnost i izvornost čvora pošiljatelja. Ulazne *SOAP* poruke koje lokalni logički čvor prima od udaljenih logičkih čvorova *Jedinica za sigurnosnu zaštitu poruka* podvrgava ispitivanju njihove vjerodostojnosti i izvornosti te uklanjanju tajnosti sadržaja. Arhitektura *Jedinice za sigurnosnu zaštitu poruka* prikazana je slikom 7.21.

Putem programskog sučelja za zaštitu poruka koje izlaže, *Jedinica za sigurnosnu zaštitu poruka* povezana je s *Usmjernikom poruka*. U slučajevima kada *Usmjernik poruka* lokalnog logičkog čvora treba uputiti zaštićenu poruku udaljenom logičkom čvoru, on pristupa sučelju za zaštitu poruka i poziva funkciju za obradu izlazne poruke. Parametri koje *Usmjernik poruka* predaje *Jedinici za sigurnosnu zaštitu poruka* su *SOAP* poruka koju je

potrebno zaštititi i ime lokalnog logičkog čvora. *Jedinica za sigurnosnu zaštitu poruka* iz zaglavlja poruke koju je potrebno zaštititi dohvaća ime izvorišnog logičkog čvora i usporeduje ga s imenom lokalnog logičkog čvora dobivenog od *Usmjernika poruka*. Ako se imena tih dvaju logičkih čvorova podudaraju, *Jedinica za sigurnosnu zaštitu poruka* zaključuje da lokalni logički čvor ima ulogu izvorišnog čvora. Sigurnosna obrada poruke na izvorišnom čvoru zahtijeva kriptiranje tijela poruke javnim ključem odredišnog čvora i digitalno potpisivanje tijela i zaglavla poruke tajnim ključem lokalnog čvora. Ako se imena izvorišnog i lokalnog logičkog čvora ne podudaraju, *Jedinica za sigurnosnu zaštitu poruka* zaključuje da lokalni logički čvor ima ulogu čvora usmjernika. Sigurnosna obrada izlazne poruke na čvoru usmjerniku zahtijeva digitalno potpisivanje dijela zaglavla poruke koji sadrži podatke o čvorovima usmjernicima. U postupku digitalnog potpisivanja tih podataka koristi se tajni ključ lokalnog logičkog čvora. *Jedinica za sigurnosnu zaštitu poruka* iz svojih unutarnjih struktura podataka dohvaća potrebne ključeve, obavlja tražene sigurnosne operacije te zaštićenu poruku vraća *Usmjerniku poruka*.



Slika 7.21 Arhitektura *Jedinice za sigurnosnu zaštitu poruka*

U slučajevima kada *Usmjernik poruka* lokalnog logičkog čvora od udaljenog logičkog čvora primi zaštićenu poruku, on pristupa sučelju za zaštitu poruka i poziva funkciju za obradu ulazne poruke. Parametri poziva funkcije za obradu ulazne poruke su zaštićena *SOAP* poruka i ime lokalnog logičkog čvora. Za vrijeme obrade ulazne poruke, iz zaglavla poruke se dohvaća podatak o imenu odredišnog logičkog čvora i uspoređuje se s imenom lokalnog logičkog čvora. Ako su imena podudarna, *Jedinica za sigurnosnu zaštitu poruka* zaključuje da lokalni logički čvor ima ulogu odredišnog čvora. Na odredišnom je logičkom čvoru potrebno provjeriti ispravnost digitalnih potpisa tijela i zaglavla poruke te dekriptirati sadržaj tijela poruke kako bi se on pretvorio u prvobitni čitljiv oblik. Provjera ispravnosti

digitalnog potpisa dijela zaglavlja koji sadrži podatke o izvorišnom i odredišnom logičkom čvoru obavlja se javnim ključem izvorišnog logičkog čvora. Provjera ispravnosti digitalnog potpisa dijela zaglavlja koji sadrži podatke o čvorovima usmjernicima obavlja se javnim ključem logičkog čvora od kojega je poruka pristigla. Dekriptiranje sadržaja tijela poruke provodi se tajnim ključem lokalnog logičkog čvora. Ako se tijekom usporedbe logičkih imena odredišnog i lokalnog logičkog čvora utvrdi postojanje razlike, zaključuje se da lokalni logički čvor obavlja ulogu čvora usmjernika. Obrada ulazne poruke na čvoru usmjerniku zahtijeva provedbu postupka provjere valjanosti digitalnog potpisa dijela zaglavlja koje se odnosi na informacije o čvorovima usmjernicima. Provjera digitalnog potpisa tih podataka obavlja se javnim ključem logičkog čvora od kojeg je poruka primljena. Dohvaćanjem odgovarajućih ključeva i obavljanjem traženih sigurnosnih operacija, obrada ulazne poruke je završena te je *Jedinica za sigurnosnu zaštitu poruka* vraća *Usmjerniku poruka*.

Unutrašnja organizacija *Jedinice za sigurnosnu zaštitu poruka* sastoji se od tri podkomponente. Središnja podkomponenta pod nazivom *Jedinica za kriptiranje i digitalno potpisivanje podataka* obavlja dvije osnovne funkcionalnosti: kriptiranje i dekriptiranje podataka te njihovo digitalno potpisivanje, odnosno provjeru valjanosti digitalnih potpisa. Ugradnja kriptiranih i digitalno potpisanih dijelova u strukturu poruke obavlja se u skladu s *WS-Security* specifikacijom.

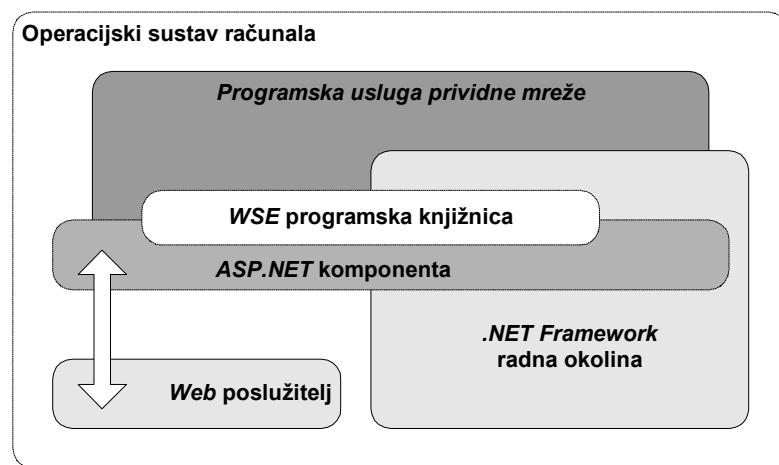
Osim *Jedinice za kriptiranje i digitalno potpisivanje podataka*, funkcionalnosti zaštite poruka obavljaju još dvije pomoćne podkomponente. *Skladište vjerodajnica* služi za dohvat tajnih i javnih ključeva lokalnih logičkih čvorova te javnih ključeva udaljenih logičkih čvorova iz spremišta vjerodajnica (engl. *trust store*) operacijskog sustava. Spremište vjerodajnica je komponenta operacijskog sustava koja osigurava siguran i pouzdan način upravljanja osjetljivim i sigurnosno visokorizičnim podacima. Tajni i javni ključevi logičkih čvorova spremjeni su u obliku *X.509* digitalnih vjerodajnica (engl. *digital certificates*) [173]. *Skladište vjerodajnica* komponente *Jedinica za sigurnosnu zaštitu poruka* dohvaća digitalne vjerodajnice iz spremišta vjerodajnica operacijskog sustava i isporučuje ih *Jedinici za kriptiranje i digitalno potpisivanje podataka*.

Dnevnik djelovanja je druga pomoćna komponenta *Jedinice za sigurnosnu zaštitu poruka*. Uloga *Dnevnika djelovanja* je vođenje podataka o djelovanju *Jedinice za kriptiranje i digitalno potpisivanje podataka*. Putem programskog sučelja za dojavu djelovanja, *Jedinica za kriptiranje i digitalno potpisivanje podataka* dojavljuje *Dnevniku djelovanja* iscrpne podatke o zahvatima provedenima nad ulaznim i izlaznim porukama, krajnjim sudionicima

komunikacije, čvorovima usmjernicima na putu između njih, vremenima poduzimanja sigurnosnih zahvata nad porukama te korištenim ključevima. Ti podaci se koriste u slučaju nastanka sigurnosnih prekršaja uzrokovanih poduzimanjem nedopuštenih radnji. Iscrpni opis programske izvedbe *Jedinice za sigurnosnu zaštitu poruka* može se pronaći u [109].

8 Programsко ostvarenje prividne mreže računalnih sustava zasnovanih na uslugama

Programska izvedba prividne mreže računalnih sustava zasnovanih na uslugama sastoji se od programskog ostvarenja algoritama i struktura podataka za uspostavljanje logičkog komunikacijskog prostora za komunikaciju programskih usluga. Tehnologije korištene za programsko ostvarenje usluge prividne mreže zasnovane su na *XML* jeziku i *Web Services* radnom okviru. *XML* jezik koristi se za definiranje struktura podataka za opisivanje logičkog komunikacijskog prostora prividne mreže. Skup tehnologija okupljenih u *Web Services* radni okvir koristi se za ostvarivanje komunikacije u logičkom komunikacijskom prostoru prividne mreže. Povezivanje logičkog i fizičkog komunikacijskog prostora postiže se povezivanjem usluge prividne mreže s operacijskim sustavom računala koji ostvaruje pristup sklopljju za fizičku mrežnu komunikaciju.



Slika 8.1 Radna okolina za izvođenje programske usluge prividne mreže

Slikom 8.1 prikazan je način povezivanja programske usluge prividne mreže s operacijskim sustavom računala i ostalim komponentama računalnog sustava. Operacijski sustav računala pruža potporu za izvođenje *web* poslužitelja i *.NET Framework* radne okoline [168] kao dviju pogonskih komponenata sustava prividne mreže. U programskom ostvarenju usluge prividne mreže, za prijenos *SOAP* poruka koristi se *HTTP* protokol. Prihvatanje poruka oblikovanih u skladu s *HTTP* protokolom obavlja *web* poslužitelj, dok obradu ugnježdenih *SOAP* poruka provode mehanizmi prividne mreže potpomognuti mehanizmima *ASP.NET* tehnologije [169]. *ASP.NET* tehnologija dio je programskog sustava *.NET Framework* razvojne i radne okoline, a namijenjena je razvoju i izvođenju primjenskih programa za *World Wide Web* uslugu. Mehanizmi *ASP.NET* tehnologije programski su

ostvareni u okviru *ASP.NET* komponente. *ASP.NET* komponenta ima uporište u *.NET Framework* radnoj okolini, a povezana je s *web* poslužiteljem. Programska usluga prividne mreže za obavljanje svojih osnovnih funkcionalnosti, kao što su oblikovanje *SOAP* poruka, njihova ugradnja u podatkovne strukture *HTTP* protokola te primanje i slanje računalnom mrežom, koristi mehanizme *ASP.NET* komponente. Funkcionalnosti svojstvene sustavu prividne mreže obavljaju algoritmi ugrađeni u programske komponente usluge prividne mreže.

Programska usluga prividne mreže i njezine sastavne komponente programski su ostvareni primjenom programskog jezika *C#* te korištenjem funkcionalnosti razreda i alata dostupnih u programskim knjižnicama *.NET Framework* radne okoline i *WSE* (engl. *Web Services Enhancements*) [170] programske knjižnice. *C#* je objektno-orientirani programski jezik visoke razine čijim se objedinjavanjem s *.NET Framework* razvojnom okolinom dobiva potpora za jednostavan i brz razvoj programskih usluga i sustava zasnovanih na uslugama. Osnovni skup funkcionalnosti za razvoj programskih usluga koji pružaju mehanizmi *ASP.NET* tehnologije proširen je razredima i alatima *WSE* programske knjižnice. *WSE* programska knjižnica nudi gotove razrede za stvaranje i obradu *SOAP* poruka, upravljanje *WS-Addressing*, *WS-Security* i ostalim proširenjima osnovnog *SOAP* protokola te mogućnost sinkrone i asinkrone komunikacije razmjenom *SOAP* poruka. Prevođenje napisanog izvornog kôda iz *C#* programskog jezika u izvodivi programski kôd programske usluge prividne mreže ostvareno je primjenom prevoditelja za *C#* programski jezik koji je dio *.NET Framework* razvojne okoline.

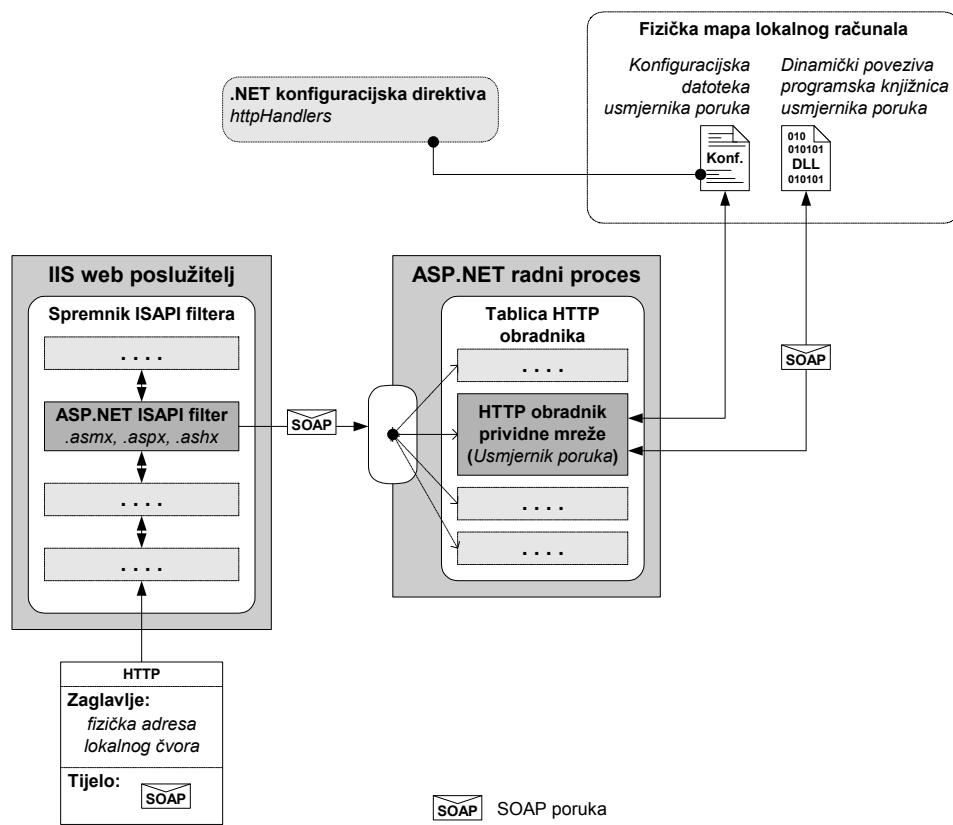
8.1 Usmjernik poruka

Usmjernik poruka je komponenta programske usluge prividne mreže zadužena za primanje i proslijedivanje *SOAP* poruka lokalnim programskim uslugama i udaljenim logičkim čvorovima. Budući da programska usluga prividne mreže za prijenos *SOAP* poruka koristi *HTTP* protokol, *Usmjernik poruka* u postupku primanja poruka koristi funkcionalnosti *IIS* (engl. *Internet Information Services*) *web* poslužitelja. U adresnom prostoru *IIS web* poslužitelja, *Usmjerniku poruka* je dodijeljena adresa fizičke pristupne točke (engl. *Uniform Resource Identifier - URI*) u obliku

`http://localhost/proizvoljna_prividna_mapa/RT.ashx`

Programska usluga koja s drugom uslugom nastoji uspostaviti komunikaciju uporabom komunikacijskog prostora prividne mreže, poruke usmjerava na pristupnu točku *Usmjernika poruka*.

Međudjelovanje *IIS web* poslužitelja i *Usmjernika poruka* prikazano je slikom 8.2. *IIS web* poslužitelj na svom ulaznom sučelju prima poruke s *HTTP* zahtjevima naslovljene fizičkom adresom *Usmjernika poruka* lokalnog logičkog čvora. Tijelo primljenog *HTTP* zahtjeva sadrži *SOAP* poruku koju treba isporučiti programskoj usluzi prividne mreže, odnosno *Usmjerniku poruka*. Obrada *HTTP* zahtjeva provodi se propuštanjem zahtjeva kroz cjevovodnu strukturu *ISAPI* (engl. *Internet Server Application Programming Interface*) [171] filtera. *ISAPI* tehnologijom omogućena je izgradnja korisničkih programa za obradu *HTTP* poruka i njihova ugradnja u radni proces *IIS web* poslužitelja. Za obradu *SOAP* poruka koristi se *ASP.NET ISAPI* filter. *ASP.NET ISAPI* filter iz tijela *HTTP* zahtjeva dohvaća *SOAP* poruke i isporučuje ih na daljnju obradu *ASP.NET* radnom procesu. Odluku o isporuci tijela *HTTP* zahtjeva *ASP.NET* radnom procesu *ASP.NET ISAPI* filter donosi na osnovi datotečnog nastavka (engl. *file extension*) sadržanog u fizičkoj adresi *HTTP* zahtjeva. *ASP.NET* radnom procesu prosljeđuju se svi zahtjevi koji su naslovljeni datotečnim nastavcima *.asmx*, *.aspx* i *.ashx*. Budući da *Usmjernik poruka* koristi fizičku adresu s *.ashx* datotečnim nastavkom, poruke naslovljene tom adresom isporučuju se na obradu *ASP.NET* radnom procesu.



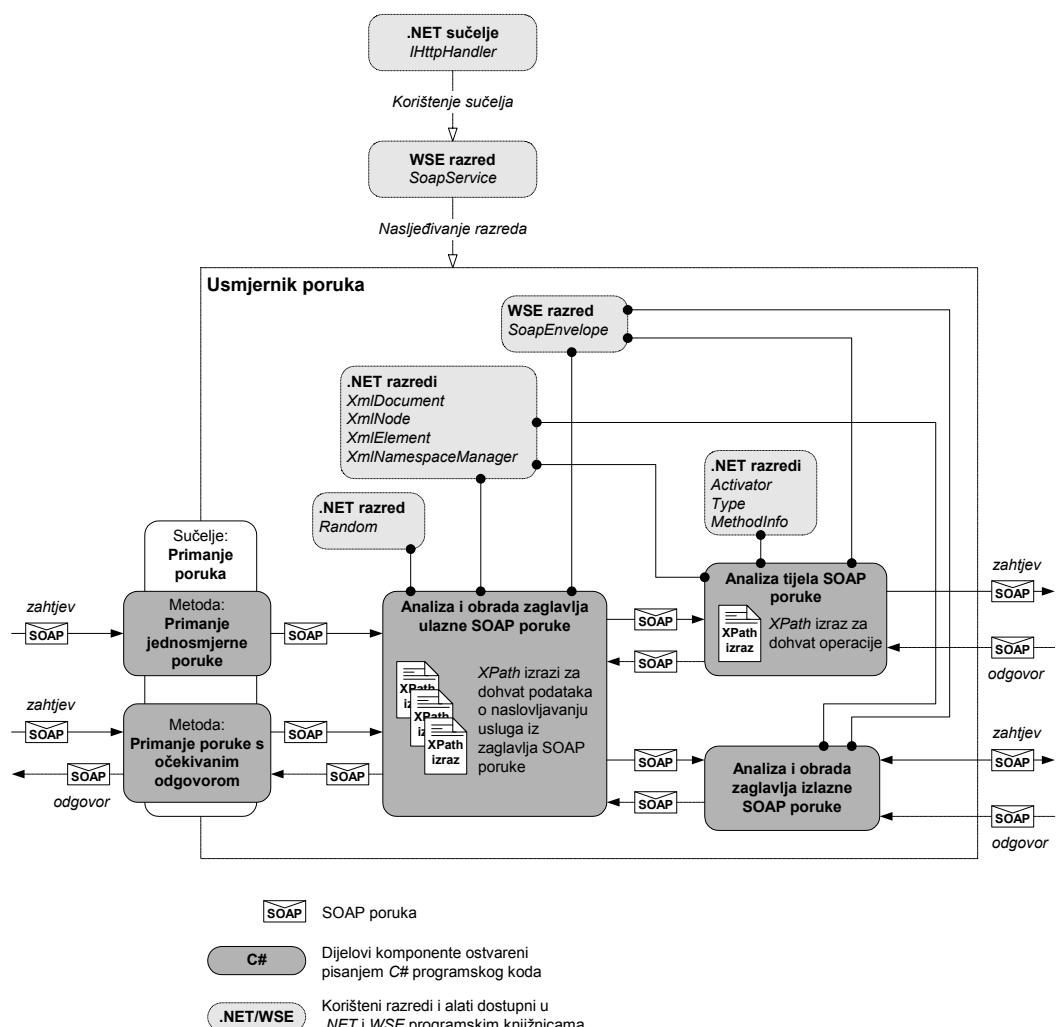
Slika 8.2 Programsко ostvarenje primanja *SOAP* poruka na sučelju *Usmjernika poruka*

Obradu *SOAP* poruka unutar *ASP.NET* radnog procesa obavlja skup *HTTP* obradnika (engl. *HTTP handler*). *HTTP* obradnici su računalni procesi za obradu *SOAP* poruka. *ASP.NET* radni proces održava tablicu *HTTP* obradnika u kojoj se *HTTP* obradnici povezuju sa *SOAP* porukama za čiju su obradu zaduženi. Za potrebe programskog ostvarenja *Usmjernika poruka* prividne mreže izgrađen je novi *HTTP* obradnik. *HTTP* obradnik prividne mreže koji obavlja funkcionalnosti *Usmjernika poruka* ostvaren je u obliku dinamički povezive programske knjižnice (engl. *dinamic link library – DLL*). Izvodivi programski kôd dostupan u dinamički povezivoj programskoj knjižnici koristi se pri obradi *SOAP* poruka upućenih na fizičku adresu *Usmjernika poruka*. Prijavljanje dinamički povezive programske knjižnice *Usmjernika poruka* u tablicu *HTTP* obradnika *ASP.NET* radnog procesa obavlja se oblikovanjem konfiguracijske datoteke. Neposredno prije isporuke *SOAP* poruke *HTTP* obradniku prividne mreže, *ASP.NET* radni proces čita postavke zapisane u konfiguracijskoj datoteci *Usmjernika poruka*. Na osnovi postavki sadržanih u konfiguracijskoj datoteci, *ASP.NET* radni proces doznaće ime datoteke koja sadrži dinamički povezivu programsku knjižnicu *Usmjernika poruka*. Konfiguracijska datoteka *Usmjernika poruka* smještena je u fizičkoj mapi računala zajedno s dinamički povezivom programskom knjižnicom. Za definiranje postavki *HTTP* obradnika u konfiguracijskoj datoteci koristi se *.NET* konfiguracijska direktiva *httpHandlers*.

Slikom 8.3 prikazano je programsko ostvarenje *Usmjernika poruka* smještenog u dinamički povezivoj programskoj knjižnici. Ključna uloga u programskom ostvarenju *HTTP* obradnika za *Usmjernik poruka* pripada dvjema programskim komponentama iz *.NET* i *WSE* programskih knjižnica. *Usmjernik poruka* ostvaren je korištenjem *.NET* programskog sučelja *IHttpHandler* i nasljeđivanjem *WSE* razreda *SoapService*. *.NET* programsko sučelje *IHttpHandler* pruža funkcionalnosti za povezivanje programskog kôda iz dinamički povezive programske knjižnice *Usmjernika poruka* s *ASP.NET* radnim procesom. *WSE* razred *SoapService* pruža osnovne funkcionalnosti za primanje *SOAP* poruka. Nasljeđivanjem razreda *SoapService* ostvarene su programske funkcionalnosti za primanje jednosmjernih *SOAP* poruka i sinkronih *SOAP* zahtjeva za koje pošiljatelji očekuju slanje povratnih *SOAP* odgovora. Dodatna obrada primljenih *SOAP* poruka svojstvena *Usmjerniku poruka* ostvarena je proširivanjem funkcionalnosti nasljeđenih iz *SoapService* razreda pisanjem *C#* programskog kôda.

Usmjernik poruka prema *ASP.NET* radnom procesu izlaže programsko sučelje koje sadrži dvije metode. Metoda za primanje jednosmjerne poruke koristi se za asinkronu komunikaciju programskih usluga. Ulazni parametar metode za primanje jednosmjerne poruke je *SOAP* poruka koju pozivajuća usluga šalje pozvanoj usluzi. Metoda za primanje

poruke s očekivanim odgovorom koristi se za sinkronu komunikaciju programskih usluga. Ulazni parametar metode za primanje poruke s očekivanim odgovorom je *SOAP* poruka sa zahtjevom koji pozivajuća usluga upućuje pozvanoj usluzi, dok je povratna vrijednost metode *SOAP* poruka s odgovorom pozvane usluge. *ASP.NET* radni proces poziva metodu *Usmjernika poruka* na osnovi vrijednosti zaglavljiva *SOAPAction* sadržanog u *SOAP* poruci. Zaglavljje *SOAPAction* je zaglavljje posebne namjene koje se koristi za povezivanje *SOAP* poruka s operacijama programskih usluga (engl. *method resolution*). Vrijednost zaglavljiva *SOAPAction* postavlja pozivajuća programska usluga putem zastupnika pozvane programske usluge tijekom upućivanja poruke lokalnom *Usmjerniku poruka*, odnosno *Zastupnik programske usluge prividne mreže udaljenog logičkog čvora* tijekom upućivanja poruke *Usmjerniku poruka* na udaljeni logički čvor.



Slika 8.3 Programsko ostvarenje *Usmjernika poruka*

Obrada *SOAP* poruka unutar *Usmjernika poruka* podijeljena je u tri cjeline. Nad svakom ulaznom *SOAP* porukom provodi se analiza i obrada sadržaja njezinih zaglavlja. Za obradu *SOAP* poruka koristi se *WSE* razred *SoapEnvelope* koji nudi funkcionalnosti za programsko rukovanje zaglavljem i tijelom *SOAP* poruke. Zaglavla primljenih *SOAP* poruka sadrže podatke o naslovljavanju programskih usluga u logičkom komunikacijskom prostoru prividne mreže. Dohvat podataka iz zaglavlja *SOAP* poruka obavlja se na osnovi ugrađenih *XPath* [172] izraza. *XPath* je programska tehnologija koja omogućava pronalaženje i dohvaćanje podataka sadržanih u *XML* dokumentima. Za dohvat podataka o naslovljavanju programskih usluga iz zaglavlja *SOAP* poruke napisana su četiri *Xpath* izraza: *XPath* izraz za dohvat logičkog imena izvorišnog logičkog čvora, *XPath* izraz za dohvat logičkog imena odredišnog logičkog čvora, *XPath* izraz za dohvat logičkog imena izvorišnog čvora usmjernika te *XPath* izraz za dohvat logičkog imena odredišnog čvora usmjernika. Za obradu podataka o naslovljavanju usluga dohvaćenih uz pomoć *XPath* izraza koriste se programske funkcionalnosti *.NET* razvojne okoline za rukovanje *XML* strukturama dostupne u razredima *XmlDocument*, *XmlNode*, *XElement* i *XmlNamespaceManager*. Korištenjem tih razreda omogućene su operacije čitanja, brisanja, izmjene i dodavanja *XML* elemenata i njihova sadržaja sadržanih u *XML* dokumentima. Za povezivanje *SOAP* zahtjeva sa *SOAP* odgovorima, *Usmjernik poruka* svakoj poruci koju obraduje pridodjeljuje jedinstvenu oznaku poruke (engl. *message ID*). Za stvaranje jedinstvene oznake poruke koriste se funkcionalnosti generatora slučajnih brojeva dostupne putem *.NET* razreda *Random*.

Na osnovi sadržaja zaglavlja ulazne *SOAP* poruke, *Usmjernik poruka* odlučuje o proslijedivanju poruke lokalnoj usluzi ili udaljenom logičkom čvoru. Ako je poruku potrebno isporučiti lokalnoj programskoj usluzi, potrebno je obaviti obradu tijela poruke. Za obradu tijela poruka koriste se funkcionalnosti *WSE* razreda *SoapEnvelope*. Iz tijela poruke potrebno je dohvatiti ime operacije pozvane programske usluge. Dohvat imena operacije obavlja se korištenjem *XPath* izraza za dohvat operacije. Nakon provjere prava pristupa do zahtijevane operacije koja se provodi u suradnji *Usmjernika poruka* i *Imenika lokalnih usluga* te pribavljanja zastupnika pozvane usluge iz *Spremnika zastupnika lokalnih usluga*, poruka se isporučuje pozvanoj usluzi. Korištenjem *.NET* razreda *Activator*, mrtvi izvodivi programski kôd zastupnika pozvane usluge uključuje se u radni proces *Usmjernika poruka*. Korištenjem razreda *Type* i *MethodInfo* obavlja se dinamičko pozivanje procedure zastupnika kojim se primljena poruka isporučuje pozvanoj programskoj usluzi.

Tijekom proslijedivanja poruke udaljenom logičkom čvoru, u zaglavljima izlazne poruke potrebno je načiniti određene izmjene u informacijama o naslovljavanju čvorova usmjernika. Ime izvorišnog logičkog čvora od kojeg je poruka primljena potrebno je

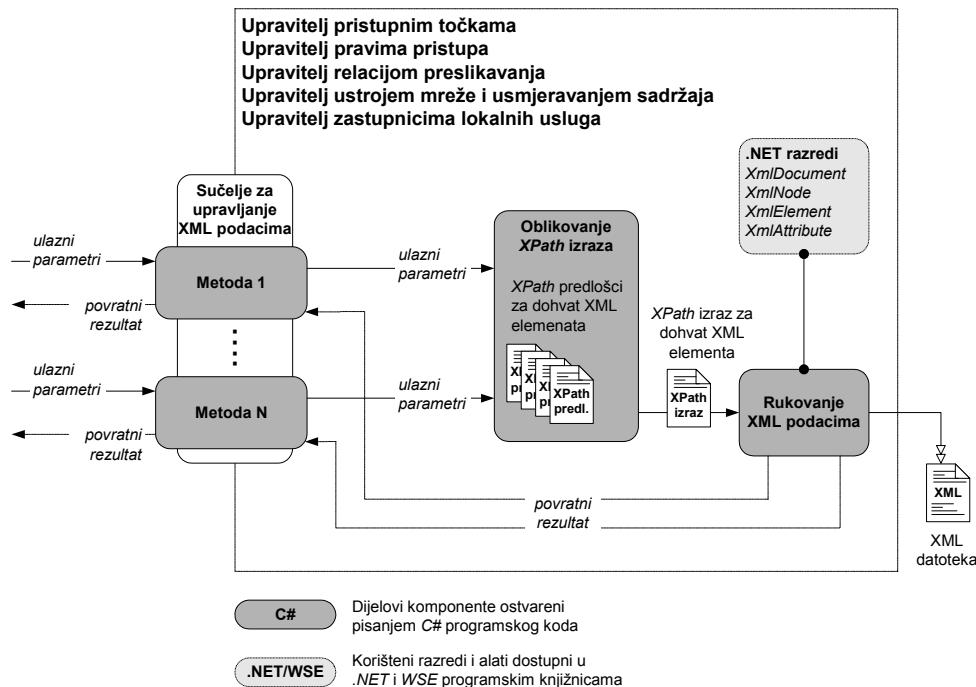
nadomjestiti imenom lokalnog logičkog čvora, dok je ime odredišnog čvora usmjernika potrebno postaviti na vrijednost logičkog imena čvora kojemu se poruka prosljeđuje. Za izmjene i ugradnju novih vrijednosti o naslovljavanju čvorova usmjernika koriste se .NET razredi *XmlDocument* , *XmlNode* , *XmlElement* i *XmlNamespaceManager* .

8.2 Imenik lokalnih usluga, Upravitelj relacijom preslikavanja i ustrojem prividne mreže i Spremnik zastupnika lokalnih usluga

Imenik lokalnih usluga, Upravitelj relacijom preslikavanja i ustrojem prividne mreže te Spremnik zastupnika lokalnih usluga su komponente prividne mreže zadužene za rukovanje tablicama sa stanjem sustava prividne mreže. Za predodžbu tablica sa stanjem sustava u radnom spremniku računala koriste se dokumenti oblikovani *XML* jezikom. Programsko ostvarenje logike tih triju komponenata sastoјi se od ostvarivanje logike za rukovanje *XML* dokumentima. Iako se programsko ostvarenje triju komponenata razlikuje u broju i vrsti programskih sučelja koje izlaže prema ostalim komponentama, ulaznim parametrima i povratnim rezultatima koji prolaze tim sučeljima te strukturi podataka u tablicama kojima rukuju, načelna programska arhitektura im je zajednička. Slikom 8.4 prikazana je načelna shema programskog ostvarenja komponenata za rukovanje *XML* tablicama. Prikazanu shemu programskog ostvarenja dijele *Upravitelj pristupnim točkama* i *Upravitelj pravima pristupa* kao podkomponente *Imenika lokalnih usluga* , *Upravitelj relacijom preslikavanja* i *Upravitelj ustrojem mreže i usmjeravanjem sadržaja* kao podkomponente *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* te *Upravitelj zastupnicima lokalnih usluga* kao podkomponenta *Spremnika zastupnika lokalnih usluga* .

Svaka komponenta za rukovanje *XML* podacima u tablicama stanja prividne mreže sastoјi se od programske cjeline za oblikovanje *XPath* izraza te programske cjeline za rukovanje *XML* podacima. Na osnovi oblikovanih *XPath* izraza dohvaćaju se podaci sadržani u tablicama stanja prividne mreže. Oblikovanje *XPath* izraza obavlja se na osnovi ugrađenih predložaka *XPath* jezika. Konačni *XPath* izrazi za dohvaćanje potrebnih vrijednosti iz *XML* dokumenata oblikuju se na osnovi vrijednosti ulaznih parametara pojedinih metoda. Oblikovani *XPath* izrazi koriste se u postupku rukovanja *XML* podacima. Programska logika za rukovanje *XML* podacima iskorištava funkcionalnosti *XmlDocument* , *XmlNode* , *XmlElement* i *XmlAttribute* razreda dostupnih iz skupa programskih knjižnica .NET radne okoline. Tablice stanja prividne mreže spremljene su u obliku *XML* datoteka. *XmlDocument* razred iz .NET programske knjižnice nudi funkcionalnosti za učitavanje i upisivanje *XML* dokumenata u datoteke. Za vrijeme pokretanja sustava prividne mreže, sve komponente

učitavaju podatke iz svojih datoteka u radni spremnik računala radi njihove učinkovitije obrade. Po završetku rada, podaci se iz radnog spremnika računala ponovno upisuju u datoteke. Upis podataka u datoteke obavlja se i nakon svake njihove izmjene.



Slika 8.4 Programsko ostvarenje komponenata za rukovanje *XML* podacima u tablicama stanja prividne mreže

Programsko ostvarenje pojedinih komponenata za rukovanje *XML* podacima razlikuje se u predlošcima *XPath* jezika koji se koriste u postupku dohvaćanja vrijednosti *XML* elemenata i algoritmima za rukovanje tablicama stanja prividne mreže. Tablicom 8.1 prikazane su vrste tablica kojima rukuju pojedine komponente te vrste *XPath* predložaka koji se pritom koriste.

Osim rukovanja podacima zapisanim u obliku *XML* dokumenata, *Spremnik zastupnika lokalnih usluga* rukuje i datotekama operacijskog sustava u koje sprema izvodivi programski kôd zastupnika lokalnih usluga. Za upravljanje datotekama zastupnika lokalnih usluga zadužen je *Ugrađivač zastupnika lokalnih usluga*, koji predstavlja podkomponentu *Spremnika zastupnika lokalnih usluga*. Programsko ostvarenje *Ugrađivača zastupnika lokalnih usluga* prikazano je slikom 8.5. *Ugrađivač zastupnika lokalnih usluga* putem svojeg programskog sučelja za ugradnju zastupnika lokalnih usluga izlaže dvije metode. Metoda za ugradnju zastupnika lokalne usluge koristi se za ugradnju izvodivog programskog kôda zastupnika u datotečni podsustav operacijskog sustava. Ulagani parametri metode za ugradnju zastupnika lokalne usluge su ime ciljnog logičkog čvora na kojem je smještena programska

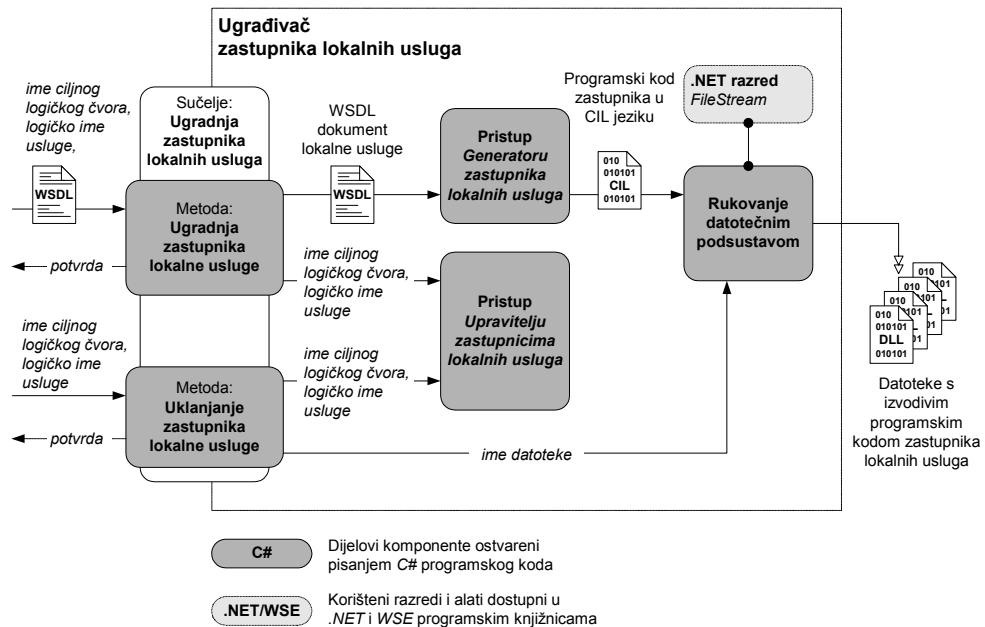
usluga za koju je potrebno ugraditi zastupnik, logičko ime programske usluge te *WSDL* dokument programske usluge. Povratna vrijednost metode je pozitivna ili negativna potvrda o uspješnosti postupka ugradnje zastupnika. Metoda za uklanjanje zastupnika lokalne usluge koristi se za brisanje izvodivog programskog kôda zastupnika iz datotečnog podsustava operacijskog sustava. Ulazni parametri metode za uklanjanje zastupnika lokalne usluge su ime ciljnog logičkog čvora na kojem je smještena programska usluga čiji je zastupnik potrebno ukloniti i logičko ime programske usluge. Povratna vrijednost metode je pozitivna ili negativna potvrda o uspješnosti postupka uklanjanja zastupnika.

Tablica 8.1 Vrste tablica i *XPath* predložaka koje koriste komponente za rukovanje stanjem prividne mreže

Vrsta komponente	Vrsta XML tablice	Skup korištenih <i>XPath</i> predložaka
Upravitelj pristupnim točkama	Tablica pristupnih točaka	Dohvat ciljnog logičkog čvora Dohvat programske usluge Dohvat adrese pristupne točke
Upravitelj pravima pristupa	Tablica prava pristupa	Dohvat ciljnog logičkog čvora Dohvat programske usluge Dohvat operacije Dohvat načina ostvarivanja prava pristupa Dohvat programskog elementa s pravom pristupa
Upravitelj relacijom preslikavanja	Tablica relacije preslikavanja	Dohvat ciljnog logičkog čvora Dohvat pravila preslikavanja Dohvat fizičke adrese čvora usmjernika
Upravitelj ustrojem mreže i usmjeravanjem sadržaja	Tablica usmjeravanja	Dohvat ciljnog logičkog čvora Dohvat pravila usmjeravanja Dohvat logičkog imena čvora usmjernika Dohvat podrazumijevanog pravila usmjeravanja
Upravitelj zastupnicima lokalnih usluga	Tablica zastupnika lokalnih usluga	Dohvat imena datoteke s programskim kôdom zastupnika lokalne usluge

Programsko ostvarenje *Ugrađivača zastupnika lokalnih usluga* sastoji se od tri programske cjeline. Logika za pristup *Generatoru zastupnika lokalnih usluga* zadužena je za proslijedivanje *WSDL* dokumenata programskih usluga do *Generatora zastupnika lokalnih usluga* te pribavljanje izvodivog programskog kôda zastupnika od *Generatora zastupnika lokalnih usluga*. Logika za pristup *Upravitelju zastupnicima lokalnih usluga* dojavljuje *Upravitelju zastupnicima lokalnih usluga* ugradnju ili uklanjanje zastupnika, kako bi ovaj na odgovarajući način zabilježio promjene u tablici zastupnika lokalnih usluga. Programska logika za rukovanje datotečnim podsustavom stvara datoteku za spremanje zastupnika lokalnih usluga te upisuje u datoteku izvodivi programski kôd pribavljen od *Generatora zastupnika lokalnih usluga*. Tijekom uklanjanja zastupnika, datoteka zadanog zastupnika se

uklanja iz datotečnog podsustava. Rukovanje datotečnim podsustavom operacijskog sustava obavlja se korištenjem funkcionalnosti *FileStream* razreda .NET programske knjižnice.



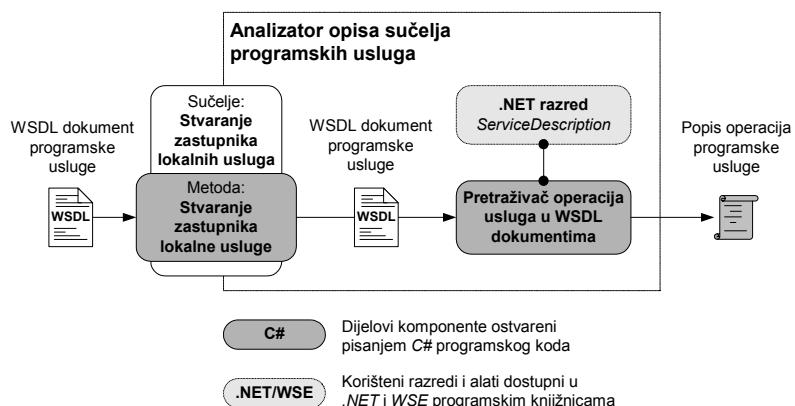
Slika 8.5 Programsко ostvarenje *Ugrađivača zastupnika lokalnih usluga*

8.3 Generator zastupnika lokalnih usluga

Generator zastupnika lokalnih usluga programski je ostvaren kao cjevododna struktura sastavljena od tri podkomponente čija je arhitektura prikazana slikom 7.14. Prva podkomponenta u cjevododnoj strukturi je *Analizator opisa sučelja programskih usluga* koji zaprima zahtjeve na ulaznom sučelju *Generatora zastupnika lokalnih usluga*. Rezultati obrade prosljeđuju se *Generatoru programskog kôda visoke razine*, koji predstavlja drugu podkomponentu unutar cjevododne strukture. Cjevododnu strukturu zaključuje *Prevoditelj programskog kôda visoke razine*, koji konačne rezultate izvođenja vraća pozivatelju koji je postavio zahtjev.

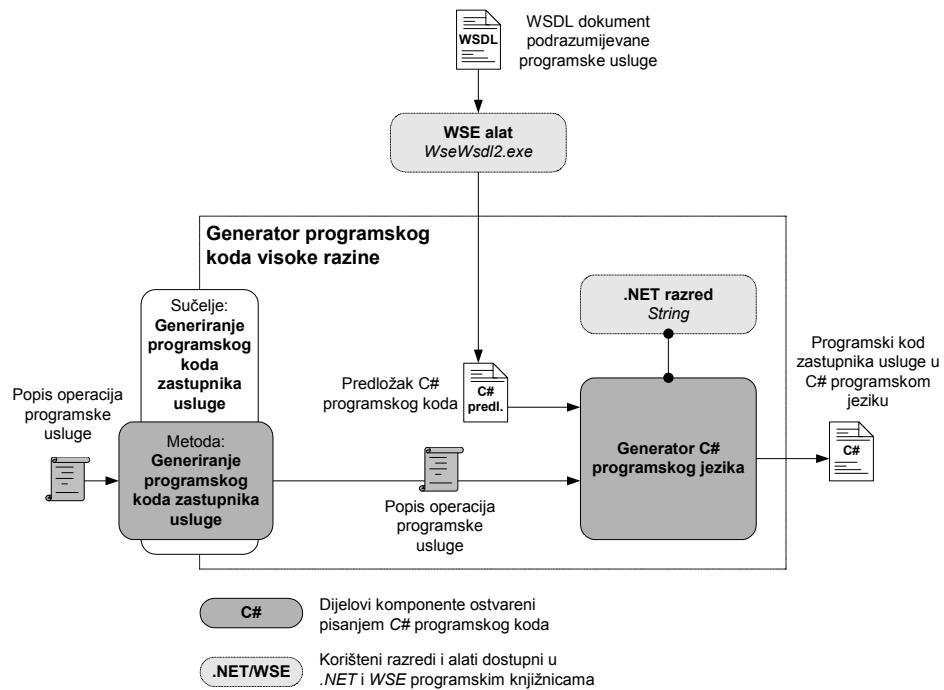
Programsko ostvarenje *Analizatora opisa sučelja programskih usluga* prikazano je slikom 8.6. *Analizator opisa sučelja programskih usluga* izlaže metodu za stvaranje zastupnika lokalne usluge kojoj se kao ulazni parametar prosljeđuje *WSDL* dokument programske usluge za koju je potrebno izgraditi zastupnik. Nad *WSDL* dokumentom provodi se postupak pretraživanja operacija koje izlaže zadana programska usluga. Rukovanje strukturama sadržanim u *WSDL* dokumentu programski je ostvareno korištenjem funkcionalnosti .NET razreda *ServiceDescription*. *ServiceDescription* razred nudi mogućnost

programskog upravljanja *WSDL* dokumentima, a u postupku programskog ostvarenja *Analizatora opisa sučelja programskih usluga* koristi se za pronađak i dohvati imena operacija programskih usluga. Na osnovi pronađenih imena operacija oblikuje se popis operacija programske usluge te se prosljeđuje drugom koraku cjevodne obrade.



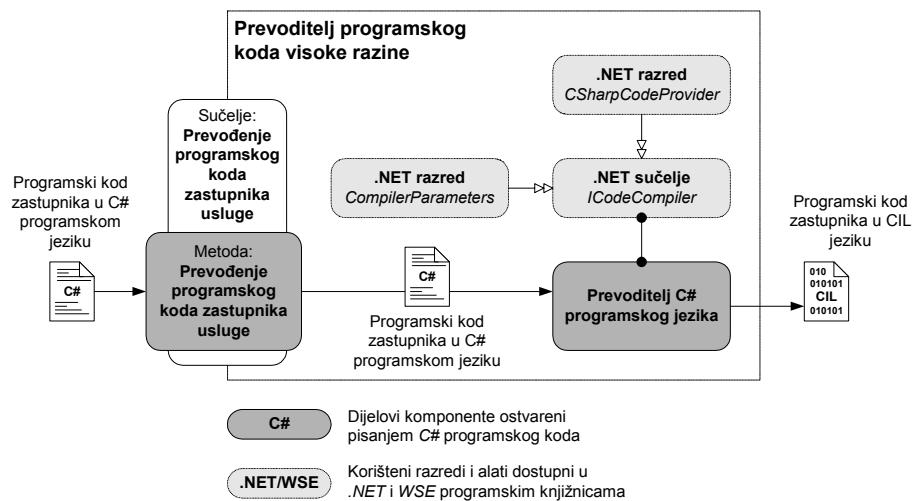
Slika 8.6 Programsko ostvarenje *Analizatora opisa sučelja programskih usluga*

U drugom koraku obrade generira se programski kôd visoke razine u programskom jeziku *C#*. Programsko ostvarenje *Generatora programskog kôda visoke razine* prikazano je slikom 8.7. *Generator programskog kôda visoke razine* izlaže programsko sučelje s metodom za generiranje programskog kôda zastupnika usluge. Ulazni parametar metode je popis operacija programske usluge. Generiranje programskog kôda u *C#* programskom jeziku obavlja se na osnovi unaprijed pripremljenog predloška *C#* programskog kôda i popisa operacija programske usluge koji se dobiva na ulazu *Generatora programskog kôda visoke razine*. Predložak *C#* programskog kôda sadrži kostur zastupnika u kojemu predviđene posebne oznake treba nadomjestiti imenima operacija zadane programske usluge. Predložak *C#* programskog kôda izgrađen je uporabom *WseWsdl2.exe* alata. *WseWsdl2.exe* je pomoći popratni alat *WSE* programske knjižnice kojim se na osnovi *WSDL* dokumenata generira programski kôd zastupnika programskih usluga u višim programskim jezicima *C#, Visual Basic, JScript ili Visual J#*. Za generiranje predloška *C#* programskog kôda iskorišten je *WSDL* dokument podrazumijevane programske usluge. U generiranom *C#* programskom kôdu, imena operacija podrazumijevane programske usluge nadomještена su posebnim oznakama koje se u postupku generiranja programskog kôda zastupnika točno određene programske usluge nadomještaju imenima operacija zadane usluge. Pretraživanje i nadomještanje posebnih oznaka imenima operacija programske usluge programski je ostvareno primjenom *.NET* razreda *String* koji nudi napredne mogućnosti rukovanja znakovnim nizovima.



Slika 8.7 Programsко ostvarenje *Generator-a programskog kôda visoke razine*

Generirani programski kôd zastupnika u C# programskom jeziku prosljeđuje se na daljnju obradu *Prevoditelju programskog kôda visoke razine*, gdje se podvrgava postupku prevodenja u naredbe CIL (engl. *Common Intermediate Language*) jezika [168]. CIL jezik je izvodivi programski kôd svojstven izvođenju od strane *spremi-i-pokreni* jezičnog procesora .NET radne okoline (engl. *Just-In-Time Compiler – JIT*) [168]. Programsko ostvarenje *Prevoditelja programskog kôda visoke razine* prikazano je slikom 8.8.



Slika 8.8 Programsko ostvarenje *Prevoditelja programskog kôda visoke razine*

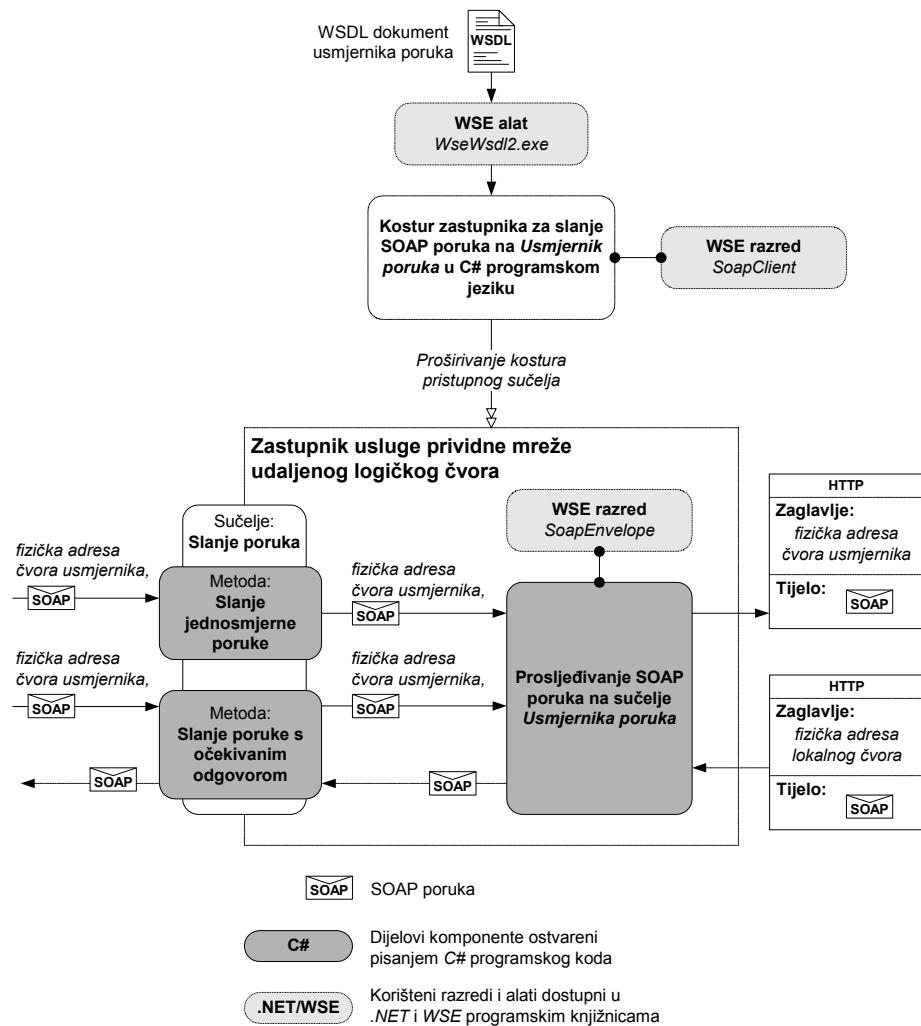
Putem programskog sučelja *Prevoditelja programskog kôda visoke razine* dostupna je metoda za prevodenje programskog kôda zastupnika usluge. Ulazni parametar metode je programski kôd zastupnika pisan u *C#* programskom jeziku. Za potrebe prevodenja *C#* programskog kôda u naredbe *CIL* jezika izgrađen je prevoditelj *C#* programskog jezika. *.NET* razvojna okolina putem svojih funkcionalnosti dostupnih u pripadajućim programskim knjižnicama nudi potporu za brzu izgradnju postupka prevodenja viših programskih jezika u naredbe *CIL* jezika. Korištenjem *.NET* programskog sučelja *ICodeCompiler* ostvaruje se programski pristup do ugrađenih prevoditelja *.NET* radne okoline. *.NET* radna okolina opremljena je s nekoliko ugrađenih prevoditelja koji različite izvorne programske jezike prevode u zajednički ciljni *CIL* jezik. Prevoditelj za *C#* programski jezik programski se koristi uporabom *.NET* razreda *CSharpCodeProvider*. Tim se razredom programsko sučelje *ICodeCompiler* podešava za pristup do ugrađenog prevoditelja za *C#* programski jezik. Definiranje postavki postupka prevodenja obavlja se uporabom razreda *CompilerParameters*. Funkcionalnostima razreda *CompilerParameters* ugrađenom se *C#* prevoditelju naređuje izgradnja optimiranog izvodivog *CIL* programa u obliku dinamički povezive programske knjižnice te mu se definiraju ugrađene *.NET* programske knjižnice koje je potrebno koristiti u postupku prevodenja.

8.4 Zastupnik usluge prividne mreže udaljenog logičkog čvora

Tijekom isporuke poruka udaljenim logičkim čvorovima, *Usmjernik poruka* koristi funkcionalnosti *Zastupnika usluge prividne mreže udaljenog logičkog čvora*. Slično kao što je isporuku poruka lokalnim uslugama potrebno obaviti putem odgovarajućih zastupnika lokalnih usluga, za isporuku poruka *Usmjerniku poruka* na udaljenom logičkom čvoru također je potrebno koristiti odgovarajući zastupnik. *Zastupnik usluge prividne mreže udaljenog logičkog čvora* koristi se za slanje *SOAP* poruka s lokalnog logičkog čvora na sučelje za primanje poruka *Usmjernika poruka* na udaljenom logičkom čvoru. Postupak programskog ostvarenja *Zastupnika usluge prividne mreže udaljenog logičkog čvora* prikazan je slikom 8.9.

U programskom ostvarenju *Zastupnika usluge prividne mreže udaljenog logičkog čvora* koristi se *WSDL* dokument *Usmjernika poruka*. Korištenjem *WSE* alata *WseWsdl2.exe* generira se kostur zastupnika za slanje *SOAP* poruka na *Usmjernik poruka*. Generirani kostur zastupnika u *C#* programskom jeziku pruža funkcionalnosti primanja *SOAP* poruka na ulaznom sučelju za slanje poruka te njihovu isporuku na fizičku adresu sadržanu u *WSDL* dokumentu *Usmjernika poruka*. Budući da *Usmjernik poruka* izlaže metode za primanje

jednosmjernih poruka i poruka s očekivanim odgovorima, generirani kostur zastupnika izlaže programsko sučelje s dvije metode za slanje tih dviju vrsta poruka. Kostur zastupnika nasljenen je iz WSE razreda *SoapClient* koji pruža osnovne funkcionalnosti za slanje sinkronih i asinkronih *SOAP* poruka ugrađenih u tijelo *HTTP* prijenosnog protokola.

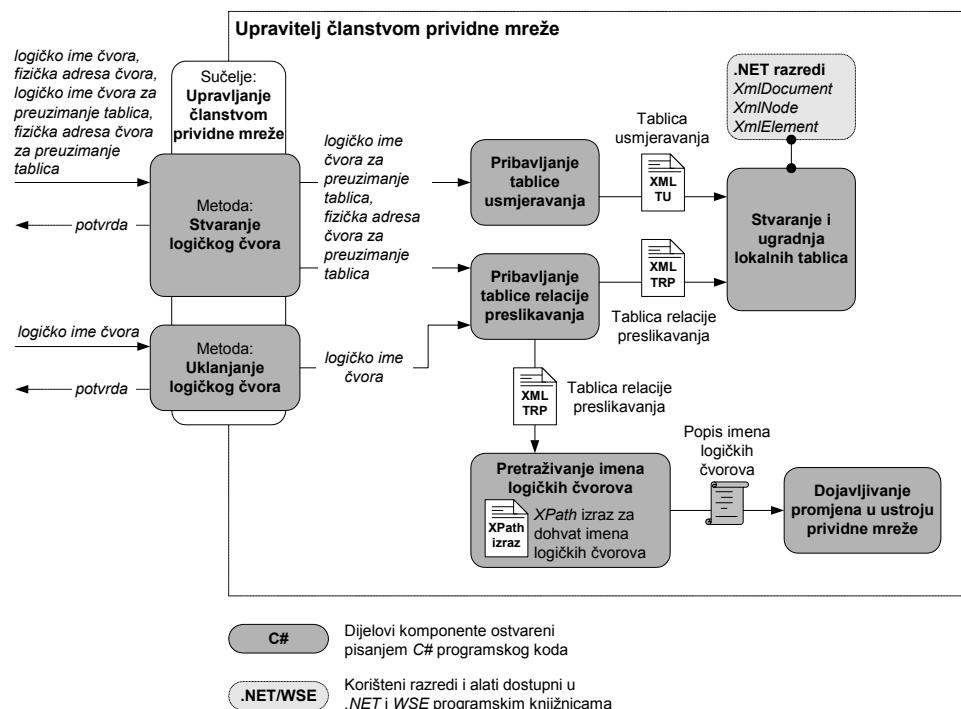


Slika 8.9 Programsко ostvarenje Zastupnika usluge prividne mreže udaljenog logičkog čvora

Postupak programskog ostvarenja *Zastupnika usluge prividne mreže udaljenog logičkog čvora* nastavlja se proširivanjem generiranog kostura zastupnika. Generirani kostur zastupnika usmjerava primljene poruke na fizičku adresu sadržanu u korištenom *WSDL* dokumentu *Usmjernika poruka*. Zbog toga je potrebno programski ostvariti mogućnost proslijedivanja poruka na zadalu fizičku adresu. Generirane metode za slanje poruka su, stoga, proširene mogućnošću zadavanja fizičke adrese udaljenog logičkog čvora kao ulaznog parametra. Za proslijedivanje primljenih *SOAP* poruka na zadane fizičke adrese, uz već spomenute funkcionalnosti *WSE* razreda *SoapClient*, koristi se i *WSE* razred *SoapEnvelope*.

8.5 Upravitelj članstvom prividne mreže

Programsko ostvarenje algoritma za održavanje potpuno povezane prividne mreže okosnica je postupka programskog ostvarenja *Upravitelja članstvom prividne mreže*. Programska arhitektura ove komponente prividne mreže sastoji se od programskog sučelja s dvije metode i pet osnovnih programskih cjelina, kao što je prikazano slikom 8.10.



Slika 8.10 Programsko ostvarenje *Upravitelja članstvom prividne mreže*

Programsko sučelje *Upravitelja članstvom prividne mreže* izlaže metodu za stvaranje logičkog čvora i metodu za uklanjanje logičkog čvora. Metoda za stvaranje logičkog čvora koristi se za postavljanje novog logičkog čvora na lokalni fizički čvor. Metoda za uklanjanje logičkog čvora koristi se za uklanjanje postojećeg logičkog čvora s lokalnog fizičkog čvora. Ulazni parametri metode za stvaranje logičkog čvora su logičko ime i fizička adresa čvora koji je potrebno uključiti u sustav prividne mreže te logičko ime i fizička adresa postojećeg čvora u sustavu prividne mreže od kojeg novopostavljeni logički čvor treba preuzeti tablice s podacima o relaciji preslikavanja i usmjeravanju sadržaja. Ulazni parametar metode za uklanjanje logičkog čvora je ime logičkog čvora koji je potrebno ukloniti iz sustava prividne mreže. Povratne vrijednosti objiju metoda su pozitivne ili negativne potvrde o uspješnosti postupka postavljanja, odnosno uklanjanja logičkog čvora sa zadanog fizičkog čvora.

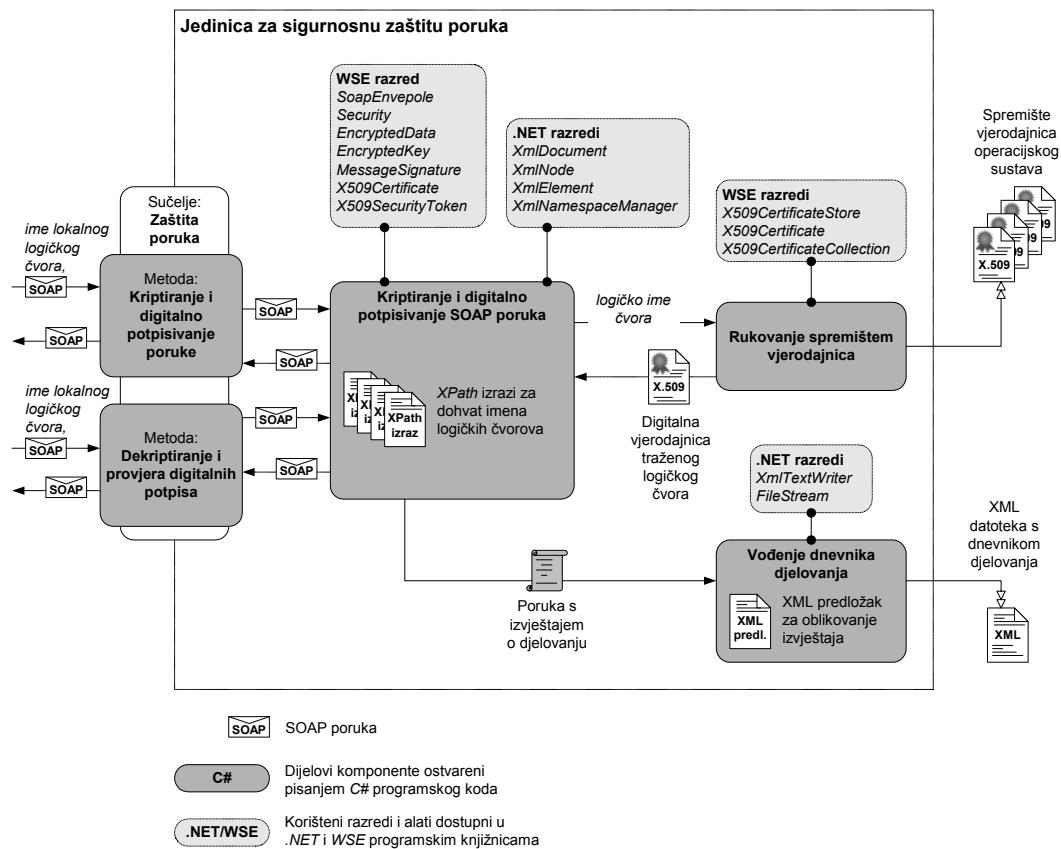
Programske celine za pribavljanje tablice relacije preslikavanja i pribavljanje tablice usmjeravanja ostvaruju programsku logiku za povezivanje sa sučeljem za upravljanje relacijom preslikavanja i ustrojem prividne mreže *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* na logičkom čvoru za preuzimanje tablica. Pribavljene tablice proširuju se pravilom preslikavanja novopostavljenog logičkog čvora i pravilom usmjeravanja sadržaja do novopostavljenog logičkog čvora te se ugrađuju u lokalni *Upravitelj relacijom preslikavanja i ustrojem prividne mreže*. Proširivanje i ugradnja tablice relacije preslikavanja i tablice usmjeravanja potpomognuti su korištenjem .NET razreda za rukovanjem XML podacima *XmlDocument*, *XmlNode* i *XmlElement*.

Nakon proširivanja i ugradnje lokalnih tablica, podatke o pravilu preslikavanja i pravilu usmjeravanja za novopostavljeni logički čvor potrebno je dojaviti svim postojećim čvorovima prividne mreže. Imena postojećih čvorova dohvaćaju se iz tablice relacije preslikavanja. Budući da ta tablica za svaki logički čvor sadrži po jedno pravilo preslikavanja, u njoj su sadržani podaci o svim logičkim čvorovima koji su članovi sustava prividne mreže. Pretraživanje tablice relacije preslikavanja i dohvati imena logičkih čvorova obavlja se na osnovi ugrađenog *XPath* izraza. Na osnovi logičkih imena dohvaćenih ugrađenim *XPath* izrazom oblikuje se popis logičkih čvorova koji koristi programska cijelina za dojavljivanje promjena u ustroju prividne mreže.

8.6 Jedinica za sigurnosnu zaštitu poruka

Jedinica za sigurnosnu zaštitu poruka programski ostvaruje zaštitu *SOAP* poruka u skladu s propisima *WS-Security* specifikacije. Putem izloženog programskog sučelja dostupne su dvije metode za sigurnosnu obradu ulaznih i izlaznih poruka. Obrada izlaznih *SOAP* poruka pokreće se pozivom metode za kriptiranje i digitalno potpisivanje poruka. Ulazne poruke podvrgavaju se sigurnosnoj obradi pozivom metode za dekriptiranje i provjeru digitalnih potpisa. Ulazni parametri metode za kriptiranje i digitalno potpisivanje poruka su *SOAP* poruka u izvornom obliku koju je potrebno kriptirati i digitalno potpisati te ime lokalnog logičkog čvora. Povratna vrijednost metode je kriptirana i digitalno potpisana *SOAP* poruka. Ulazni parametri metode za dekriptiranje i provjeru digitalnih potpisa su kriptirana i digitalno potpisana *SOAP* poruka te ime lokalnog logičkog čvora. Ako lokalni logički čvor ima ulogu odredišnog logičkog čvora, povratna vrijednost metode je *SOAP* poruka u izvornom obliku. Ako lokalni logički čvor ima ulogu čvora usmjernika, povratna vrijednost metode ovisi o ispravnosti digitalnih potpisa logičkog čvora od kojeg je poruka primljena. Ako poruka sadrži valjane digitalne potpise, povratna vrijednost metode je *SOAP*

poruka istovjetna ulaznoj *SOAP* poruci. Ako digitalni potpisi sadržani u poruci nisu ispravni, povratna vrijednost metode je oznaka sigurnosne pogreške.



Slika 8.11 Programsко ostvarenje Jedinice za sigurnosnu zaštitu poruka

Programsko ostvarenje *Jedinice za sigurnosnu zaštitu poruka* prikazano je slikom 8.11. Programska arhitektura je podijeljena na tri programske cjeline. Središnji dio programske arhitekture *Jedinice za sigurnosnu zaštitu poruka* pripada *Jedinici za kriptiranje i digitalno potpisivanje SOAP poruka*. Operacije kriptiranja provode se nad tijelom *SOAP* poruke, dok se operacije digitalnog potpisivanja provode nad tijelom i zaglavljem poruke. Osnovicu programskog ostvarenja *Jedinice za kriptiranje i digitalno potpisivanje SOAP* poruka predstavlja programska potpora za provedbu *WS-Security* standarda, ugrađena u *WSE* programsku knjižnicu. Razred *SoapEnvelope* koristi se kao osnovni razred za obradu *SOAP* poruka. Razred *Security* koristi se za rukovanje zaglavljima *SOAP* poruke kojima se osnovni *SOAP* protokol proširuje elementima *WS-Security* standarda. Kriptiranje tijela poruke provodi se uporabom *EncryptedData* razreda. Taj razred pruža mogućnost kriptiranja tijela poruke zadanim ključevima te ugradnju kriptiranih dijelova u strukturu *SOAP* poruke. Ugradnja ključeva korištenih u postupku kriptiranja u strukturu *SOAP* poruke provedena je

uporabom razreda *EncryptedKey*. Digitalno potpisivanje zaglavlja poruke te ugradnju digitalnih potpisa u strukturu poruke provodi razred *MessageSignature*. U postupku digitalnog potpisivanja podataka te u postupku kriptiranja ključeva korištenih za kriptiranje tijela poruke koriste se javni ključevi primatelja poruka sadržani u digitalnim vjerodajnicama. *Jedinica za sigurnosnu zaštitu poruka* koristi digitalne vjerodajnice u obliku *X.509* certifikata [173]. Pri rukovanju digitalnim vjerodajnicama koriste se funkcionalnosti razreda *X509Certificate*, dok se njihova ugradnja u strukturu *SOAP* poruke provodi iskorištavanjem funkcionalnosti razreda *X509SecurityToken*. Za pomoćne funkcionalnosti pri ugradnji sigurnosnih elemenata *WS-Security* standarda u strukturu *SOAP* poruke koriste se razredi *XmlElement*, *XmlNode* i *XmlDocument*, kao osnovni razredi .NET programske knjižnice za rukovanje *XML* podacima. Za dohvati podataka iz tijela *SOAP* poruke koje je potrebno podvgnuti postupku kriptiranja i dekriptiranja koristi se razred *SoapEnvelope*. Dohvat imena logičkih čvorova iz zaglavlja poruke kojima je potrebno pridružiti odgovarajuće digitalne potpisane podatke provodi se na osnovu ugrađenih *XPath* izraza. U tu su svrhu napisana četiri *Xpath* izraza: *XPath* izraz za dohvati logičkog imena izvorišnog logičkog čvora, *XPath* izraz za dohvati logičkog imena odredišnog logičkog čvora, *XPath* izraz za dohvati logičkog imena izvorišnog čvora usmjernika te *XPath* izraz za dohvati logičkog imena odredišnog čvora usmjernika.

Digitalne vjerodajnice logičkih čvorova koje se koriste u postupku sigurnosne obrade *SOAP* poruka spremljene su u spremištu vjerodajnica (engl. *trust store*) operacijskog sustava računala. Rukovanje spremištem vjerodajnica programski je ostvareno iskorištavanjem funkcionalnosti *WSE* razreda *X509CertificateStore*, *X509Certificate* i *X509CertificateCollection*. Razredom *X509CertificateStore* ostvaruje se programski pristup do skladišta vjerodajnica. Budući da skladište vjerodajnica dopušta pridruživanje većeg broja digitalnih vjerodajnica istom logičkom čvoru, za njihovo se dohvaćanje koriste funkcionalnosti razreda *X509CertificateCollection*. Odabir jedne od raspoloživih vjerodajnica i isporuka odabrane vjerodajnice *Jedinici za kriptiranje i digitalno potpisivanje SOAP* poruka obavlja se uz pomoć razreda *X509Certificate*.

Djelovanje *Jedinice za kriptiranje i digitalno potpisivanje SOAP* poruka bilježi se u *Dnevniku djelovanja* u obliku *XML* datoteke. *Jedinica za kriptiranje i digitalno potpisivanje SOAP* poruka oblikuje tekstualne poruke s izvještajima o djelovanju te ih šalje *Dnevniku djelovanja*. Na osnovi podataka u porukama s izvještajima o djelovanju oblikuju se *XML* dokumenti koji se upisuju u datoteku dnevnika. Pretvorba poruka s izvještajima o djelovanju u *XML* dokumente za upis u datoteku dnevnika provodi se na osnovi ugrađenog predloška *XML* dokumenta. Za popunjavanje *XML* predloška podacima o izvršenom djelovanju koristi

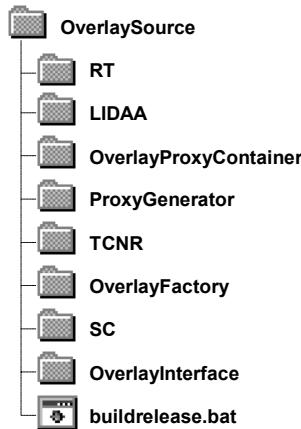
se .NET razred *XmlTextWriter*, koji omogućava jednostavnu ugradnju tekstualnih poruka u strukture *XML* dokumenata. Razred *FileStream* koristi se za pristup datoteci s dnevnikom djelovanja.

8.7 Prevođenje i postavljanje programske usluge prividne mreže

Izvorni programski kôd programske usluge prividne mreže napisan u *C#* programskom jeziku, zajedno s popratnim podacima koji se koriste u postupku prevođenja u ciljni program, smješten je u datotečni podsustav operacijskog sustava prema slici 8.12. Vršna mapa *OverlaySource* sadrži osam podmapa u kojima se nalaze datoteke s izvornim kôdom pojedinih komponenata. Mapa *RT* sadrži datoteke s izvornim programom *Usmjernika poruka* i *Zastupnika usluge prividne mreže udaljenog logičkog čvora*. U mapi *LIDAA* nalaze se datoteke s izvornim programom *Imenika lokalnih usluga*. Mapa *OverlayProxyContainer* sadrži datoteke s izvornim programom *Spremnika zastupnika lokalnih usluga*, dok su datoteke s izvornim programom *Generatorka zastupnika lokalnih usluga* smještene u mapi *ProxyGenerator*. Datoteke s izvornim programom *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* smještene su u mapi *TCNR*. Mapa *OverlayFactory* sadrži datoteke s izvornim programom *Upravitelja članstvom prividne mreže*, a mapa *SC* datoteke s izvornim programom *Jedinice za sigurnosnu zaštitu poruka*. U mapi *OverlayInterface* nalaze se datoteke s izvornim programom pristupnog sučelja usluge prividne mreže. U tim je datotekama programski ostvareno javno izlaganje sučelja pojedinih komponenata usluge prividne mreže u skladu s propisima *WebServices* radnog okvira.

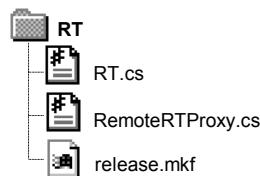
Osim podmapa s datotekama izvornog programa pojedinih komponenata usluge prividne mreže, vršna mapa *OverlaySource* sadrži datoteku *buildrelease.bat* s naredbama za prevođenje izvornog programa svake pojedine komponente u izvodivi program usluge prividne mreže. Pri prevođenju izvornog programa u izvodivi program koristi se program *nmake*. Program *nmake* je pomoći alat .NET Framework razvojne okoline koji na osnovi naredbi za prevođenje programa sadržanih u ulaznoj datoteci prevodi zadani izvorni u zadani ciljni program. Prevođenje izvornog programskega kôda usluge prividne mreže u izvodivi program ostvaruje se pokretanjem datoteke *buildrelease.bat*. U toj su datoteci napisane naredbe ljske operacijskog sustava koje redom pozivaju program *nmake* za prevođenje svake pojedine komponente. Pri svakom pozivu programa *nmake* kao ulazni parametar predaje se datoteka *release.mkf* za odgovarajuću komponentu usluge prividne mreže. Svaka komponenta u vlastitoj mapi ima smještenu datoteku *release.mkf* koja sadrži naredbe za

njeno prevodenje iz izvornog u ciljni program u skladu sa sintaksnim pravilima programa *nmake*.



Slika 8.12 Organizacija izvornog programskega koda programske usluge prividne mreže

Organizacija izvornog programskega koda *Usmjernika poruka* i *Zastupnika usluge prividne mreže udaljenog logičnega čvora* u mapi *RT* prikazana je slikom 8.13. Datoteka *RT.cs* sadrži programski kód *Usmjernika poruka* u C# programskom jeziku. Datoteka *RemoteRTProxy.cs* sadrži programski kód *Zastupnika usluge prividne mreže udaljenog logičnega čvora* u C# programskom jeziku. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kód iz datoteka *RT.cs* i *RemoteRTProxy.cs* prevodi u izvodivi CIL program. Postupak prevodenja ostvaruje se pozivanjem prevoditelja za C# programski jezik ugrađenog u .NET Framework razvojnu okolinu. Poziv ugrađenog prevoditelja za C# programski jezik ostvaruje se pozivanjem programa *csc.exe* s odgovarajućim parametrima.



Slika 8.13 Organizacija izvornog programskega koda *Usmjernika poruka* i *Zastupnika usluge prividne mreže udaljenog logičnega čvora*

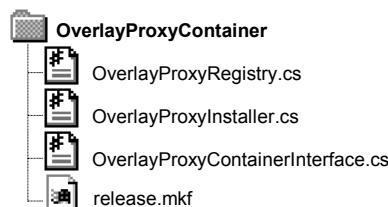
Organizacija izvornog programskega koda *Imenika lokalnih usluga* u mapi *LIDAA* prikazana je slikom 8.14. Datoteka *LocalInterfaceDirectory.cs* sadrži programski kód *Upravitelja pristupnim točkama* u C# programskom jeziku. Datoteka *AuthorizationManager.cs* sadrži programski kód *Upravitelja pravima pristupa* u C# programskom jeziku. Datoteka *LIDAAInterface.cs* sadrži programski kód u C# programskom

jeziku za povezivanje programskih sučelja *Imenika lokalnih usluga* na odgovarajuća sučelja *Upravitelja pristupnim točkama* i *Upravitelja pravima pristupa*. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *LocalInterfaceDirectory.cs*, *AuthorizationManager.cs* i *LIDAAInterface.cs* prevodi u izvodivi *CIL* program pozivanjem ugrađenog prevoditelja za C# programski jezik.



Slika 8.14 Organizacija izvornog programskog kôda *Imenika lokalnih usluga*

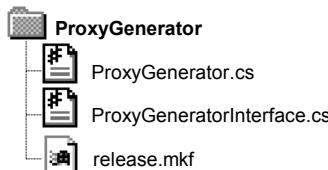
Organizacija izvornog programskog kôda *Spremnika zastupnika lokalnih usluga* u mapi *OverlayProxyContainer* prikazana je slikom 8.15. Datoteka *OverlayProxyRegistry.cs* sadrži programski kôd *Upravitelja zastupnicima lokalnih usluga* u C# programskom jeziku. Datoteka *OverlayProxyInstaller.cs* sadrži programski kôd *Ugrađivača zastupnika lokalnih usluga* u C# programskom jeziku. Datoteka *OverlayProxyContainerInterface.cs* sadrži programski kôd u C# programskom jeziku za povezivanje programskih sučelja *Spremnika zastupnika lokalnih usluga* na odgovarajuća sučelja *Upravitelja zastupnicima lokalnih usluga* i *Ugrađivača zastupnika lokalnih usluga*. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *OverlayProxyRegistry.cs*, *OverlayProxyInstaller.cs* i *OverlayProxyContainerInterface.cs* prevodi u izvodivi *CIL* program pozivanjem ugrađenog prevoditelja za C# programski jezik.



Slika 8.15 Organizacija izvornog programskog kôda *Spremnika zastupnika lokalnih usluga*

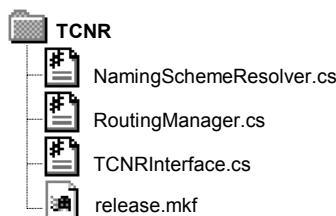
Organizacija izvornog programskog kôda *Generatora zastupnika lokalnih usluga* u mapi *ProxyGenerator* prikazana je slikom 8.16. Datoteka *ProxyGenerator.cs* sadrži programski kôd *Analizatora opisa sučelja programskih usluga*, *Generatora programskega kôda visoke razine* i *Prevoditelja programskega kôda visoke razine* u C# programskom jeziku.

Datoteka *ProxyGeneratorInterface.cs* sadrži programski kôd u C# programskom jeziku za povezivanje programskog sučelja *Generatora zastupnika lokalnih usluga* na sučelje *Analizatora opisa sučelja programskih usluga*. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *ProxyGenerator.cs* i *ProxyGeneratorInterface.cs* prevodi u izvodivi *CIL* program pozivanjem ugrađenog prevoditelja za C# programski jezik.



Slika 8.16 Organizacija izvornog programskog kôda *Generatora zastupnika lokalnih usluga*

Organizacija izvornog programskog kôda *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* u mapi *TCNR* prikazana je slikom 8.17. Datoteka *NamingSchemeResolver.cs* sadrži programski kôd *Upravitelja relacijom preslikavanja* u C# programskom jeziku. Datoteka *RoutingManager.cs* sadrži programski kôd *Upravitelja ustrojem mreže i usmjeravanjem sadržaja* u C# programskom jeziku. Datoteka *TCNRIInterface.cs* sadrži programski kôd u C# programskom jeziku za povezivanje programskih sučelja *Upravitelja relacijom preslikavanja i ustrojem prividne mreže* na odgovarajuća sučelja *Upravitelja relacijom preslikavanja* i *Upravitelja ustrojem mreže i usmjeravanjem sadržaja*. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *NamingSchemeResolver.cs*, *RoutingManager.cs* i *TCNRIInterface.cs* prevodi u izvodivi *CIL* program pozivanjem ugrađenog prevoditelja za C# programski jezik.



Slika 8.17 Organizacija izvornog programskog kôda *Upravitelja relacijom preslikavanja i ustrojem prividne mreže*

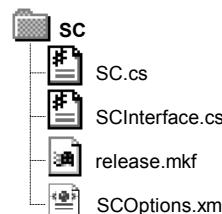
Organizacija izvornog programskog kôda *Upravitelja članstvom prividne mreže* u mapi *OverlayFactory* prikazana je slikom 8.18. Datoteka *OverlayFactory.cs* sadrži

programski kôd *Upravitelja članstvom prividne mreže* u C# programskom jeziku s izvedbom algoritma održavanja potpuno povezane prividne mreže. Datoteka *OverlayInterfaceProxy.cs* sadrži zastupnik sučelja usluge prividne mreže u C# programskom jeziku kojim se ostvaruje pristup do *Upravitelja članstvom prividne mreže* na udaljenim logičkim čvorovima. Datoteka *OverlayFactoryInterface.cs* sadrži programski kôd u C# programskom jeziku s izvedbom programskog sučelja *Upravitelja članstvom prividne mreže*. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *OverlayFactory.cs*, *OverlayInterfaceProxy.cs* i *OverlayFactoryInterface.cs* prevodi u izvodivi CIL program pozivanjem ugrađenog prevoditelja za C# programski jezik.



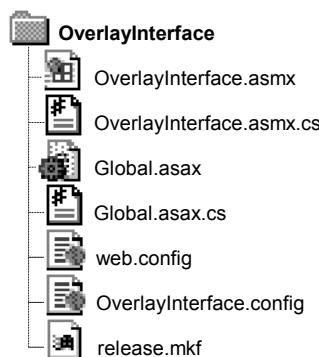
Slika 8.18 Organizacija izvornog programskog kôda *Upravitelja članstvom prividne mreže*

Organizacija izvornog programskog kôda *Jedinice za sigurnosnu zaštitu poruka* u mapi *SC* prikazana je slikom 8.19. Datoteka *SC.cs* sadrži programski kôd *Jedinice za sigurnosnu zaštitu poruka* u C# programskom jeziku s izvedbom algoritma za kriptiranje i digitalno potpisivanje SOAP poruka. Datoteka *SCInterface.cs* sadrži programski kôd u C# programskom jeziku s izvedbom programskog sučelja *Jedinice za sigurnosnu zaštitu poruka*. Datoteka *SCOptions.xml* sadrži upute koje *Jedinica za sigurnosnu zaštitu poruka* koristi za vrijeme rada. U toj datoteci su upisane informacije o potrebi vođenja dnevnika djelovanja, imenu datoteke s dnevnikom djelovanja, imenu spremišta vjerodajnica operacijskog sustava iz kojeg se dohvaćaju i u koje se spremaju X.509 certifikati i slično. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *SC.cs* i *SCInterface.cs* prevodi u izvodivi CIL program pozivanjem ugrađenog prevoditelja za C# programski jezik.



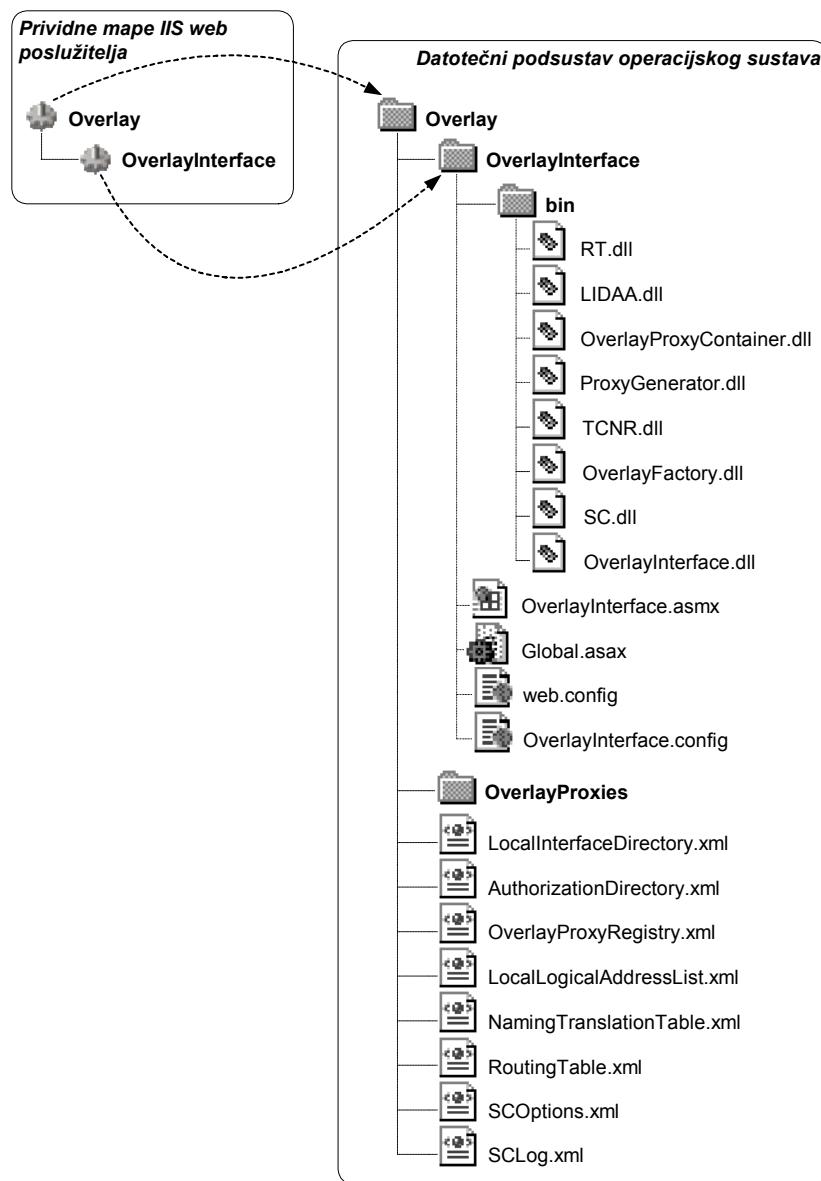
Slika 8.19 Organizacija izvornog programskog kôda *Jedinice za sigurnosnu zaštitu poruka*

Povezivanje programskih sučelja pojedinih komponenata na javna sučelja programske usluge prividne mreže u skladu s propisima *Web Services* radnog okvira ostvareno je u mapi *OverlayInterface*. Organizacija izvornog programskog kôda kojim su ostvarena sučelja usluge prividne mreže prikazan je slikom 8.20. Datoteka *OverlayInterface.asmx.cs* sadrži programski kôd javnih pristupnih sučelja programske usluge prividne mreže u C# programskom jeziku. Datoteka *OverlayInterface.asmx* sadrži upravljačke upute kojima se ASP.NET radnom procesu navodi ime datoteke *HTTP* obradnika koji je zadužen za obradu *SOAP* poruka upućenih programskoj usluzi prividne mreže. Datoteka *Global.asax.cs* sadrži programski kôd u C# programskom jeziku koji ostvaruje funkcionalnosti koje se izvode pri pokretanju i zaustavljanju programske usluge prividne mreže. Programska usluga prividne mreže pri pokretanju učitava podatke iz XML datoteka sa stanjem sustava prividne mreže u radni spremnik računala. Pri zaustavljanju usluge prividne mreže, podaci sa stanjem sustava se iz radnog spremnika računala zapisuju u XML datoteke. Datoteka *Global.asax* sadrži upravljačke upute kojima se ASP.NET radnom procesu navodi ime datoteke *HTTP* obradnika koji je zadužen za provedbu funkcionalnosti ostvarenih u datoteci *Global.asax.cs*. Datoteka *web.config* je sustavna konfiguracijska datoteka programske usluge prividne mreže. Sadrži podatke koje koristi ASP.NET radni proces neposredno prije isporuke *SOAP* poruka *HTTP* obradniku prividne mreže. U datoteci *web.config* sadržani su podaci o pravima pristupa programskoj usluzi prividne mreže, o potrebi bilježenja korištenja usluge, o datoteci u koju se upisuju zapisi o korištenju i slično. Datoteka *OverlayInterface.config* je primjenska konfiguracijska datoteka programske usluge prividne mreže. U datoteku *OverlayInterface.config* upisani su podaci u imenima datoteka s tablicama stanja prividne mreže. Datoteka *release.mkf* sadrži naredbe za program *nmake* kojima se programski kôd iz datoteka *OverlayInterface.asmx.cs* i *Global.asax.cs* prevodi u izvodivi *CIL* program pozivanjem ugrađenog prevoditelja za C# programski jezik.



Slika 8.20 Organizacija izvornog programskog kôda pristupnih sučelja programske usluge prividne mreže

Prevodenje programske usluge prividne mreže iz programskog jezika C# u izvodivi CIL jezik obavlja se pokretanjem datoteke *buildrelease.bat*. Iz datoteke *buildrelease.bat* pokreće se slijedno prevodenje svake komponente prividne mreže pozivom programa *nmake*. Ulazni parametar programu *nmake* je datoteka *release.mkf* koja odgovara komponenti prividne mreže čiji je postupak prevodenja u tijeku. Program za prevodenje napisan u datoteci *buildrelease.bat* gradi datotečnu strukturu prema slici 8.21.



Slika 8.21 Postavljanje programske usluge prividne mreže

Vršna mapa *Overlay* sadrži podmape *OverlayInterface* i *OverlayProxies*. Mapa *OverlayInterface* sadrži datoteke s izvodivim programima programske usluge prividne mreže. U podmapi *bin* smještene su dinamički povezive programske knjižnice u obliku *.dll*

datoteka za svaku od komponenata programske usluge prividne mreže. Osim podmape *bin*, mapa *OverlayInterface* sadrži upravljačke datoteke *OverlayInterface.asmx* i *Global.asax*, sustavnu konfiguracijsku datoteku *web.config* te primjensku konfiguracijsku datoteku *OverlayInterface.config*. Mapa *OverlayProxies* predviđena je za spremanje datoteka s izvodivim programskim kôdom zastupnika lokalnih programskih usluga kojima upravlja *Spremnika zastupnika lokalnih programskih usluga*.

Osim podmapa *OverlayInterface* i *OverlayProxies*, vršna mapa *Overlay* sadrži i *XML* datoteke sa stanjem sustava prividne mreže. Datoteka *LocalInterfaceDirectory.xml* sadrži tablicu pristupnih točaka. Datoteka *AuthorizationDirectory.xml* sadrži tablicu prava pristupa. Datoteka *OverlayProxyRegistry.xml* sadrži popis zastupnika lokalnih programskih usluga. Podaci o imenima logičkih čvorova koji su postavljeni na lokalni čvor fizičke mreže upisuju se u datoteku *LocalLogicalAddressList.xml*. Tablica relacije preslikavanja upisuje se u datoteku *NamingTranslationTable.xml*. Tablica usmjeravanja upisuje se u datoteku *RoutingTable.xml*. Radne postavke *Jedinice za sigurnosnu zaštitu poruka* upisane su u datoteku *SCOptions.xml*, dok se datoteka *SCLog.xml* koristi kao dnevnik djelovanja.

Postavljanje (engl. *deployment*) programske usluge prividne mreže sastoji se od stvaranja prividnih mapa u radnom prostoru *IIS web* poslužitelja i njihovim povezivanjem na odgovarajuće fizičke mape datotečnog podsustava operacijskog sustava računala. Postupak podešavanja *IIS web* poslužitelja za pravilan rad programske usluge prividne mreže prikazan je slikom 8.21. U radnom prostoru *IIS web* poslužitelja potrebno je stvoriti vršnu prividnu mapu *Overlay*, a unutar nje prividnu podmapu *OverlayInterface*. Prividne mape *Overlay* i *OverlayInterface* potrebno je preslikati na istoimene fizičke mape datotečnog podsustava. U tako podešenom poslužitelju, adresa programske usluge prividne mreže je

`http://localhost/Overlay/OverlayInterface/OverlayInterface.asmx`

dok je adresa *Usmjernika poruka*

`http://localhost/Overlay/OverlayInterface/RT.ashx`

9 Zaključak

Računarstvo zasnovano na uslugama približilo je metodologiju razvoja raspodijeljenih računalnih sustava krajnjim korisnicima mreže Internet. Raspodijeljeni sustavi zasnovani na uslugama grade se iskorištavanjem postojećih programskih usluga dostupnih u mreži i njihovim povezivanjem u nove usluge. Povezivanje programskih usluga korisnicima omogućava izgradnju raspodijeljenih aplikacija prilagođenih trenutnim potrebama i raspodijeljene okoline za izvođenje tih aplikacija. Osnovna pretpostavka računarstva zasnovanog na uslugama je primjena otvorenih i standardima usvojenih programskih tehnologija za izlaganje i korištenje programskih usluga. Otvorenim i standardnim programskim tehnologijama prevladavaju se tehnološke raznovrsnosti i raznorodnosti njihove programske izvedbe. Okosnicu računarstva zasnovanog na uslugama čini skup otvorenih i standardima usvojenih programskih tehnologija za objedinjavanje programske potpore okupljen u *Web Services* radni okvir.

Izgradnja i izvođenje raspodijeljenih aplikacija zasnovanih na uslugama olakšani su primjenom posredničkog sustava. Prividna raspodijeljena računalna okolina je raspodijeljeni posrednički sustav koji pruža potporu za izgradnju, postavljanje i izvođenje raspodijeljenih aplikacija zasnovanih na uslugama. Razvijen je u suradnji Zavoda za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu i Instituta za telekomunikacije tvrtke Ericsson Nikola Tesla d.d. iz Zagreba. Prividnu raspodijeljenu računalnu okolinu čine prividna mreža za komunikaciju programskih usluga, podsustav za otkrivanje i postavljanje programskih usluga, podsustav za prevodenje i izvođenje raspodijeljenih programa te podsustav za siguran pristup uslugama.

U ovom magistarskom radu definirana je arhitektura te je oblikovana i programski ostvarena prividna mreža računalnih sustava zasnovanih na uslugama. Prividna mreža računalnih sustava zasnovanih na uslugama je raspodijeljeni programski sustav za komunikaciju programskih usluga koji olakšava oblikovanje, izgradnju i održavanje raspodijeljenih aplikacija zasnovanih na uslugama te podiže razinu njihove sigurnosti, učinkovitosti i otpornosti na kvarove. Primjenom prividne mreže, nad fizičkom se mrežom uspostavlja logički komunikacijski prostor s korisnički definiranim sustavom naslovljavanja mrežnih čvorova i programskih usluga te sustavom logičkog usmjeravanja poruka. Raspodjelom programskih usluga u logičkom komunikacijskom prostoru prividne mreže, uz uporabu logičkih imena mrežnih čvorova i programskih usluga, omogućena je izgradnja raspodijeljenih aplikacija neosjetljivih na promjene uvjeta u fizičkim radnim okolinama. Time prividna mreža pruža potporu prenosivosti raspodijeljenih aplikacija zasnovanih na

uslugama u radne okoline s promjenjivim brojem mrežnih čvorova i njihovih mrežnih adresa. Sustav logičkog naslovljavanja mrežnih čvorova i programske usluge pruža potporu za izgradnju raspodijeljenih aplikacija zasnovanih na uslugama koje su otporne na kvarove mrežnih čvorova. Dinamičkim izmjenama relacije preslikavanja logičkih imena mrežnih čvorova u njihove mrežne adrese omogućeno je premještanje programske usluge s pokvarenih na ispravne mrežne čvorove za vrijeme rada aplikacije. Osim otpornosti na kvarove mrežnih čvorova, mogućnost premještanja programske usluge bez utjecaja na izvođenje izgrađenog raspodijeljenog sustava pruža potporu za izgradnju mehanizama učinkovitog upravljanja opterećenjem mrežnih čvorova. Sustav logičkog usmjeravanja poruka omogućava definiranje logičkog ustroja prividne mreže na postojećoj infrastrukturi fizičke mreže. Logički ustroj prividne mreže omogućava usmjeravanje poruka na primjenskoj razini, čime se podiže razina sigurnosti i privatnosti korisnika i pružatelja programske usluge.

Oblikovanje i programska izvedba sustava prividne mreže slijedi smjernice arhitekture zasnovane na uslugama. Sustav prividne mreže programski je ostvaren u obliku programske usluge primjenom *Web Services* radnog okvira programskih tehnologija za izgradnju sustava zasnovanih na uslugama. Zbog dobre potpore za izgradnju programskih sustava zasnovanih na uslugama, za izgradnju i izvođenje programskog sustava prividne mreže koriste se programski jezik *C#*, programske knjižnice i razvojni alati *.NET Framework* razvojne i radne okoline te *ASP.NET* tehnologija. Prošireni skup standarda iz osnovnog *Web Services* radnog okvira u obliku *WS-Addressing* i *WS-Security* specifikacija primjenjen je uporabom *Web Services Enhancements* programske knjižnice. Pristup programskoj usluzi prividne mreže ostvaruje se *SOAP* protokolom. Prijenos *SOAP* poruka računalnom mrežom obavlja se *HTTP* protokolom. Razvojna okolina *.NET Framework* nudi programske naredbe kojima se prevoditelju programskog jezika *C#* naređuje automatsko proširivanje izvodivog programskog kôda programske usluge programskim kôdom za njezino izlaganje putem *HTTP* protokola. Programska usluga prividne mreže dostupna *HTTP* protokolom ugrađena je u *Internet Information Server* web poslužitelj koji se izvodi pod okriljem *Microsoft Windows* operacijskih sustava.

Funkcionalna ispravnost izgrađenog sustava prividne mreže ispitana je u zatvorenim radnim okružnjima lokalnih mreža te u otvorenom okruženju globalne mreže Internet. Tijekom ispitivanja rada sustava, korisnicima je omogućeno dodavanje i uklanjanje logičkih čvorova u i iz sustava prividne mreže te korištenje oblikovane prividne mreže za izgradnju i izvođenje raspodijeljenih aplikacija zasnovanih na uslugama. Izgrađeni sustav prividne mreže pokazao se naročito korisnim pri prenošenju izgrađenih raspodijeljenih aplikacija iz

radne okoline globalne mreže Internet u radnu okolinu lokalne mreže s promjenjenim adresama mrežnih čvorova i manjim brojem raspoloživih računala. Ispitivanjem rada sustava prividne mreže pod povećanim opterećenjem pokazana je zavisnost algoritma održavanja potpuno povezane prividne mreže o broju logičkih čvorova i broju istodobno postavljenih zahtjeva za uključivanjem i isključivanjem čvorova iz mreže.

Postavljanjem sustava prividne mreže u radnu okolinu te ispitivanjem radnih svojstava u zatvorenim i otvorenim mrežnim okruženjima stvorene su prepostavke za daljnja istraživanja i nadogradnju sustava novim funkcionalnostima. Jedno od značajnih proširenja je izgradnja programirljivog sustava za sigurnu komunikaciju programskih usluga koji korisnici prividne mreže mogu prilagođavati vlastitim potrebama. Osim toga, proširenjem sigurnosnog sustava elementima za mjerjenje učinkovitosti komunikacijskog sustava te praćenje razine sigurnosnog rizika moguće je ostvariti samoprilagodljiv sigurnosni sustav koji u danim radnim uvjetima optimira omjer sigurnosti i učinkovitosti sustava.

10 Literatura

- [1] P. Van Roy, S. Haridi: “**Concepts, Techniques, and Models of Computer Programming**”, MIT Press, 2004
- [2] I. Sommerville: “**Software Engineering**”, 6th Edition, Adison-Wesley, 2001
- [3] R. S. Pressman: “**Software Engineering: A Practicioner’s Approach**”, 5th Edition, McGraw-Hill, 2001
- [4] M. P. Singh, M. N. Huhns: “**Service-Oriented Computing: Semantics, Processes, Agents**”, John Wiley & Sons, 2005
- [5] M. Shaw, D. Garlan: “**Software Architecture: Perspectives of an Emerging Discipline**”, Prentice-Hall, 1996
- [6] A.S. Tanenbaum, M.V. Steen: “**Distributed Systems: Principles and Paradigms**”, Prentice Hall, 2002
- [7] A.S. Tanenbaum: “**Distributed Operating Systems**”, Prentice Hall, 1995
- [8] Q.H. Mahmoud (Editor): “**Middleware for Communications**”, John Wiley & Sons, 2004
- [9] A. S. Tanenbaum: “**Computer Networks**”, 4th Edition, Prentice Hall, 2003
- [10] M. Huhns, M. P. Singh: “**Service-Oriented Computing: Key Concepts and Principles**”, IEEE Internet Computing, Vol. 9, No. 1, January/February 2005, pp. 75-81
- [11] M. P. Papazoglou, D. Georgakopoulos: “**Service-Oriented Computing**”, Communications of the ACM, Vol. 46, No. 10, October 2003, pp. 25-28
- [12] M. P. Papazoglou: “**Service-Oriented Computing: Concepts, Characteristics and Directions**”, 4th IEEE International Conference on Web Information Systems Engineering (WISE 2003), December 2003, Roma, Italy
- [13] M. P. Papazoglou: “**Extending the Service-Oriented Architecture**”, Business Integration Journal, February 2005, pp. 18-21
- [14] L. Dimitriou: “**Distributed Parallel Computing with Web Services**”, Web Services Journal, Vol. 5, No. 2, February 2005, pp. 28-31
- [15] J.-Y. Chung, K.-J. Lin, R.G. Mathieu: “**Web Services Computing: Advancing Software Interoperability**”, Computer, vol. 36, no. 10, October 2003, pp. 35-37
- [16] D. Linthicum: “**What Level is Your SOA**”, Web Services Journal, Vol. 4, No. 12, December 2004, pp. 18-19
- [17] M. Turner, D. Budgen, P. Brereton: “**Turning Software Into a Service**”, Computer, Vol. 36, No. 10, October 2003, pp. 38-44
- [18] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau: “**Extensible Markup Language (XML) 1.0**”, 3rd Edition, W3C Recommendation, February 2004
<http://www.w3.org/TR/REC-xml>
- [19] E. Bertino, E. Ferrari: “**XML and Data Integration**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp 75 – 76

- [20] M. Gudgin, M. Hadley, N. Mendelsohn, J.J. Moreau, H.F. Nielsen: “**SOAP Version 1.2 Part 1: Messaging Framework**”, W3C Recommendation, June 2003
<http://www.w3.org/TR/soap12-part1>
- [21] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana: “**Web Service Description Language (WSDL) 1.1**”, W3C Note, March 2001
<http://www.w3.org/TR/wsdl>
- [22] T. Bellwood, et al.: “**UDDI Version 3.0**”, UDDI Spec TC, July 2002
<http://uddi.org/pubs/uddi-v3.00-published-20020719.htm>
- [23] R. Sessions: “**Fuzzy Boundaries: Objects, Components, and Web Services**”, ACM Queue, Vol. 2, No. 9, December 2004, pp. 40-47
- [24] C. S. Yeo, R. Buyya, H. Pourreza, R. Eskicioglu, P. Graham, F. Sommers: “**Cluster Computing: High-Performance, High-Availability, and High-Throughput Processing on a Network of Computers**”, Handbook of Innovative Computing, Albert Zomaya (Editor), Springer Verlag, 2005
- [25] F. Berman, G. C. Fox, A. J. G. Hey (Editors): “**Grid Computing: Making the Global Infrastructure a Reality**”, John Wiley & Sons, 2003
- [26] D. Niculescu: “**Communication Paradigms for Sensor Networks**”, IEEE Communications Magazine, Vol. 43, No. 3, March 2005, pp. 116-122
- [27] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci: “**A Survey on Sensor Networks**”, IEEE Communications Magazine, Vol. 40, No. 8, August 2002, pp. 102-114
- [28] C.-Y. Chong, S. P. Kumar: “**Sensor Networks: Evolution, Opportunities, and Challenges**”, Proceedings of the IEEE, Vol. 91, No. 8, August 2003, pp. 1247-1256
- [29] L. W. McKnight, S. Bradner: “**Wireless Grids**”, IEEE Internet Computing, Vol. 8, No. 4, July/August 2004, pp. 24-31
- [30] H. Sumino, N. Ishikawa, T. Kato, J. Hjelm, S. Murakami, K. Miyatsu: “**Design and Implementation of a Multicast Instant Messaging System on the Mobile P2P Network**”, Proceedings of the First International Conference on Mobile Computing and Ubiquitous Networking (ICMU 2004), January 2004, Yokosuka, Japan
- [31] N. Ishikawa, T. Kato, H. Sumino, O. Takahashi, J. Hjelm, K. Miyatsu, S. Murakami: “**Jupiter: Peer-to-Peer Networking Platform toward Ubiquitous Communications**”, NTT DoCoMo Inc. Japan, Ericsson Research Japan, Ericsson Research Sweden
- [32] A. Crespo, H. Garcia-Molina: “**Semantic Overlay Networks for P2P Systems**”, Technical Report, Computer Science Department, Stanford University, October 2002
- [33] D. M. Russel: “**UbiComp 2003: Sensors in Seattle**”, IEEE Pervasive Computing, Vol. 3, No. 2, April 2004, pp. 76-80
- [34] G. Bell: “**Intimate Computing?**”, IEEE Internet Computing, Vol. 8, No. 6, November/December 2004, pp. 91-93
- [35] C. Thompson: “**Everything Is Alive**”, IEEE Internet Computing, Vol. 8, No. 1, January/February 2004, pp. 83-86

- [36] J. Myllymaki, J. Kaufman: “**High-Performance Spatial Indexing for Location-Based Services**”, Proceedings of the 12th International Conference on World Wide Web, 2003, Budapest, Hungary, pp. 112-117
- [37] B. Schneier: “**Applied Cryptography: Protocols, Algorithms, and Source Code in C**”, 2nd Edition, John Wiley & Sons, 1995
- [38] M. Abadi, A. Birrell, T. Wobber: “**Access Control in a World of Software Diversity**”, University of California, Santa Cruz, and Microsoft Research
- [39] D. Kesdogan, J. Egner, R. Buschkes: “**Stop-And-Go-MIXes Providing Probabilistic Anonymity in an Open System**”, Proceedings of the International Information Hiding Workshop, April 1998, pp. 83-98
- [40] D. M. Goldschlag, M. G. Reed, P. F. Syverson: “**Hiding Routing Information**”, First International Workshop on Information Hiding, Springer-Verlag, May 1996, pp. 137–150
- [41] D. Chaum: “**The Dining Cryptographers Problem: Unconditional Sender Untraceability**”, Journal of Cryptology, Vol. 1, No. 1, 1988, pp. 65-75
- [42] D. Chaum: “**Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms**”, Communications of the ACM, Vol. 24, No. 2, February 1981, pp. 84-88
- [43] Electronic Frontier Foundation: “**Tor: Overview**”
<http://tor.eff.org/overview.html>
- [44] L. Gong: “**Peer-to-Peer Networks in Action**”, IEEE Internet Computing, Vol. 6, No. 1, January/February 2002, pp. 37-39
- [45] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. Balakrishnan: “**Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications**”, Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, September 2001, San Diego, California, USA, pp. 149-160
- [46] K. Aberer, M. Punceva, M. Hauswirth, R. Schmidt: “**Improving Data Access in P2P Systems**”, IEEE Internet Computing, Vol. 6, No. 1, January/February 2002, pp. 57-67
- [47] I. Clarke, S. G. Miller, T. W. Hong, O. Sandberg, B. Wiley: “**Protecting Free Expression Online with Freenet**”, IEEE Internet Computing, Vol. 6, No. 1, January/February 2002, pp. 40-49
- [48] J. Kubiatowicz, D. Bindel, et al.: “**OceanStore: An Architecture for Global-Scale Persistent Storage**”, Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), November, 2000, Cambridge, Massachusetts, USA, pp. 190-201
- [49] M. Ripeanu, A. Iamnitchi, I. Foster: “**Mapping the Gnutella Network**”, IEEE Internet Computing, Vol. 6, No. 1, January/February 2002, pp. 50-57
- [50] B. Y. Zhao, J. Kubiatowicz, A. D. Joseph: “**Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing**”, Berkeley, Computer Science Division, University of California, 2001
- [51] R. Stern: “**Napster: A Walking Copyright Infringement?**”, IEEE Micro, Vol. 20, No. 6, November/December 2000, pp. 4-5, 95

- [52] H.-C. Hsiao, C.-T. King: “**Tornado: A Capability-Aware Peer-to-Peer Storage Overlay**”, Journal of Parallel and Distributed Computing, Vol. 64, No. 6, June 2004, pp. 747-758
- [53] Y. Amir, C. Danilov: “**Reliable Communication in Overlay Networks**”, International Conference on Dependable Systems and Networks (DSN'03), June 2003, San Francisco, California, USA, pp. 511-520
- [54] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker: “**A Scalable Content-Addressable Network**”, Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, September 2001, San Diego, California, USA, pp. 161-172
- [55] W. Nejdl, W. Siberski, M. Wolpers, C. Schmitz: “**Routing and Clustering in Schema-Based Super Peer Networks**”, Submitted to the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), February 2003, Berkeley, California, USA
- [56] Y. Huang, S. N. Bhatti: “**Decentralized Resilient Grid Resource Management Overlay Networks**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 372-379
- [57] M. Yoon, Y. Shin: “**On-Demand One-Time Overlay Multicast Tree for Multimedia Transmission over Large Internet Environment**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 541-546
- [58] A. Rowstron, P. Druschel: “**Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems**”, Proceedings of the 18th FIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), November 2001, Heidelberg, Germany, pp. 329-350
- [59] G. Ciaccio: “**NEBLO: Anonymity in a Structured Overlay**”, Technical Report, University of Genova, Italy, May 2005
- [60] D. Andersen, H. Balakrishnan, F. Kaashoek, R. Morris: “**Resilient Overlay Networks**”, Proceedings of 18th ACM Symposium on Operating Systems Principles (SOSP 2001), October 2001, Banff, Canada
- [61] T. Fuhrmann: “**A Self-Organizing Routing Scheme for Random Networks**”, Proceedings of the 4th IFIP-TC6 Networking Conference, May 2005, Waterloo, Canada, pp. 1366-1370
- [62] J. Wu, I. Stojmenovic: “**Ad Hoc Networks**”, Computer, Vol. 37, No. 2, February 2004, pp. 29-31
- [63] Marco Conti, Gaia Maselli, Giovanni Turi, and Silvia Giordano: “**Cross-Layering in Mobile Ad Hoc Network Design**”, Computer, Vol. 37, No. 2, February 2004, pp. 48-51
- [64] N. Milanovic, M. Malek, A. Davidson, V. Milutinovic: “**Routing and Security in Mobile Ad Hoc Networks**”, Computer, Vol. 37, No. 2, February 2004, pp. 61-65
- [65] Q. Xue, A. Ganz: “**Ad Hoc QoS On-Demand Routing (AQOR) in Mobile Ad Hoc Networks**”, Journal of Parallel and Distributed Computing, Vol. 63, No. 2, February 2003, pp. 154-165
- [66] T. Goff, N. Abu-Ghazaleh, D. Phatak, R. Kahvecioglu: “**Preemptive Routing in Ad Hoc Networks**”, Journal of Parallel and Distributed Computing, Vol. 63, No. 2, February 2003, pp. 123-140

- [67] L. Venkatraman, D. P. Agrawal: “**Strategies for Enhancing Routing Security in Protocols for Mobile Ad Hoc Networks**”, Journal of Parallel and Distributed Computing, Vol. 63, No. 2, February 2003, pp. 214-227
- [68] M. A. Menasce: “**MOM vs. RPC**”, IEEE Internet Computing, Vol. 9, No. 2, March/April 2005, pp. 90-93
- [69] M. Brambilla, S. Ceri, M. Passamani, A. Riccio: “**Managing Asynchronous Web Services Interactions**”, IEEE International Conference on Web Services (ICWS 2004), June 2004, San Diego, CA, USA, pp. 80-87
- [70] J. Verzulli: “**Concurrently Accessing Multiple Web Service Instances**”, Web Services Journal, Vol. 2, No. 3, March 2002, pp. 60-63
- [71] R. Figueiredo, P. A. Dinda, J. Fortes: “**Resource Virtualization Renaissance**”, Computer, Vol. 38, No. 5, May 2005, pp. 28-31
- [72] R. Figueiredo, P. A. Dinda, J. Fortes: “**Virtual Distributed Environments in a Shared Infrastructure**”, Computer, Vol. 38, No. 5, May 2005, pp. 63-69
- [73] L.-J. Zhang, H. Li, H. Lam: “**Toward a Business Process Grid for Utility Computing**”, IT Professional, Vol. 6, No. 5, September/October 2004, pp. 62-64
- [74] S. Murphy, J. Reddy: “**Delivering, Administrating, & Tracking Distributed Web Services**”, Web Services Journal, Vol. 4, No. 12, December 2004, pp. 34-37
- [75] R. Peng, K. A. Hua, G. L. Hamza-Lup: “**A Web Services Environment for Internet Scale Sensor Computing**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 101-108
- [76] G. Hamilton: “**Evolving the Java Platform**”, Software Development Times, No. 126, BZ Media LLC, May 2005, pp. 33-34
- [77] N. Holmes: “**The Internet, the Web, and the Chaos**”, Computer, Vol. 38, No. 9, September 2005, pp. 106-108
- [78] R. Harper: “**Toward a New Communications Genre**”, Computer, Vol. 38, No. 8, August 2005, pp. 89-91
- [79] R. Altman: “**Demystifying the ESB**”, Web Services Journal, Vol. 5, No. 2, February 2005, pp. 32-34
- [80] D. Chappell: “**ESB Myth Busters**”, Web Services Journal, Vol. 5, No. 2, February 2005, pp. 22-26
- [81] J. Kobiels: “**Enterprise Service Bus Products**”, Web Services Journal, Vol. 4, No. 12, December 2004, pp. 8-8
- [82] C. Boulton: “**Sun, Others Prep Java Integration Spec**”, October 2004
<http://www.internetnews.com/dev-news/article.php/3427751>
- [83] T. Tryzbiak: “**Easy as Apple Pie**”, Web Services Journal, Vol. 2, No. 3, March 2002, pp. 44-46
- [84] M. Lehmann: “**Deploying Large-Scale Interoperable Web Services Infrastructure**”, Web Services Journal, Vol. 5, No. 1, January 2005, pp. 10-15
- [85] T. Erl: “**A Look Ahead to the Service-Oriented World**”, an excerpt from “**Service-Oriented Architecture: Concepts and Technology**” by Thomas Erl, April 2005

- <http://wldj.sys-con.com/read/48928.htm>
- [86] X. Yang, M. Hayes, A. Usher, M. Spivack: “**Developing Web Services in a Computational Grid Environment**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 600-603
- [87] D. Talia: “**The Open Grid Services Architecture: Where the Grid Meets the Web**”, IEEE Internet Computing, Vol. 6, No. 6, November 2002, pp. 67-71
- [88] K. Arnold: “**Programmers Are People, Too**”, ACM Queue, Vol. 3, No. 5, June 2005, pp. 54-59
- [89] A. Milanović: “**Programski model zasnovan na uslugama**”, doktorska disertacija u pripremi, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
- [90] A. Milanovic, S. Srbljic, D. Skrobo, D. Capalija, S. Reskovic: “**Coopetition Mechanisms for Service-Oriented Distributed Systems**”, Proceedings of the 3rd International Conference on Computing, Communications and Control Technologies, July 2005, Austin, Texas, USA, Vol. 1, pp. 118-123
- [91] S. Rešković: “**Sinkronizacijski i komunikacijski mehanizmi za sustave zasnovane na uslugama**”, diplomski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, rujan 2005.
- [92] D. Čapalija: “**Objava-preplata mehanizmi za ostvarivanje mreža zasnovanih na sadržaju**”, diplomski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, lipanj 2005.
- [93] I. Gavran: “**Korisnički jezik programskog modela zasnovanog na uslugama**”, magistarski rad u pripremi, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
- [94] D. Skrobo: “**Raspodijeljeno usporedno interpretiranje programa u arhitekturama zasnovanim na uslugama**”, magistarski rad u pripremi, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
- [95] D. Skrobo, A. Milanovic, S. Srbljic: “**Distributed Program Interpretation in Service-Oriented Architectures**”, Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics, July 2005, Orlando, Florida, USA, Vol. 4, pp. 193-197
- [96] T. Andrews, et al.: “**Business Process Execution Language for Web Services (BPEL4WS)**”, version 1.1, Microsoft, IBM, Siebel Systems, BEA, and SAP, May 2003
<http://www-128.ibm.com/developerworks/library/specification/ws-bpel>
- [97] M. Podravec: “**Otkrivanje i postavljanje usluga u sustavima zasnovanim na uslugama**”, magistarski rad u pripremi, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
- [98] M. Podravec, I. Skuliber, S. Srbljic: “**Service Discovery and Deployment in Service-Oriented Computing Environment**”, Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics, July 2005, Orlando, Florida, USA, Vol. 3, pp. 5-10

- [99] M. Popović: “**Nadziranje pristupa računalnim sustavima zasnovanim na uslugama**”, magisterski rad u pripremi, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
- [100] M. Popovic, I. Benc, S. Srbljic: “**Access Control in Hierarchies of Protected Cells**”, Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics, July 2005, Orlando, Florida, USA, Vol. 3, pp. 356-361
- [101] T. Berners-Lee, L. Masinter, M. McCahill: “**Uniform Resource Locators (URL)**”, RFC 1738, December 1994
<http://www.ietf.org/rfc/rfc1738.txt>
- [102] J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: “**Hypertext Transfer Protocol - HTTP/1.1**”, RFC 2616, June 1999
<http://www.ietf.org/rfc/rfc2616.txt>
- [103] J. B. Postel: “**Simple Mail Transfer Protocol (SMTP)**”, RFC 821, August 1982
<http://www.ietf.org/rfc/rfc0821.txt>
- [104] J. Postel, J. Reynolds: “**File Transfer Protocol (FTP)**”, RFC 959, October 1985
<http://www.ietf.org/rfc/rfc959.txt>
- [105] D. Box, et al.: “**Web Services Addressing (WS-Addressing)**”, W3C Member Submission, August 2004
<http://www.w3.org/Submission/ws-addressing>
- [106] A. Nadalin, C. Kaler, P. Hallam-Baker, R. Monzillo: “**Web Services Security: SOAP Message Security 1.0**”, OASIS Standard 200401, March 2004
- [107] D. Eastlake, J. Reagle, D. Solo, M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon: “**XML-Signature Syntax and Processing**”, W3C Recommendation, February 2002
<http://www.w3.org/TR/xmldsig-core>
- [108] T. Imamura, B. Dillaway, E. Simon: “**XML Encryption Syntax and Processing**”, W3C Recommendation, December 2002
<http://www.w3.org/TR/xmlenc-core>
- [109] T. Čohar: “**Sigurnosni mehanizmi u logičkim komunikacijskim mrežama s ravnopravnim sudionicicima**”, diplomska rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, lipanj 2005.
- [110] B. W. Lampson: “**Computer Security in the real World**”, Computer, Vol. 37, No. 6, June 2004, pp. 37-46
- [111] M. Andrews, J. A. Whittaker: “**Computer Security**”, IEEE Security & Privacy, Vol. 2, No. 5, September 2004, pp. 68-71
- [112] H. Kim, S. Bahk: “**Real-Time Visualization of Network Attacks on High-Speed Links**”, IEEE Network, Vol. 18, No. 5, September/October 2004, pp. 30-39
- [113] N. McCullagh: “**Securing E-Mail with Identity-Based Encryption**”, IT Professional, Vol. 7, no. 3, May/June 2005, pp. 61-64
- [114] J. Viega: “**Security: Problem Solved?**”, ACM Queue, Vol. 3, No. 5, June 2005, pp. 40-50

- [115] E. Damiani, P. Samarati, S. De Capitani di Vimercati, S. Paraboschi: “**Controlling Access to XML Documents**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 18-28
- [116] J. Zhang: “**Trustworthy Web Services: Actions for Now**”, IT Professional, Vol. 7, No. 1, January/February 2005, pp. 32-36
- [117] M. O’Neill: “**Securing Web Services**”, Web Services Journal, Vol. 2, No. 3, March 2002, pp. 20-27
- [118] J. Ducharme: “**Making Second Generation Web Services Secure**”, Web Services Journal, Vol. 2, No. 3, March 2002, pp. 48-53
- [119] A. Avižienis, J. C. Laprie, B. Randell, C. Landwehr: “**Basic Concepts and Taxonomy of Dependable and Secure Computing**”, IEEE Transactions on Dependable and Secure Computing, Vol. 1, No. 1, January/March 2004, pp. 11-30
- [120] S. Durchslag: “**Beyond the Hype...the Reality of Early Web Services Adoption**”, Web Services Journal, Vol. 2, No. 3, March 2002, pp. 28-31
- [121] J. Bloomberg: “**Building Security Into a Service-Oriented Architecture**”, ZapThink LLC, May 2003
<http://www.zapthink.com/report.html?id=WP-0112>
- [122] K. Czajkowski, et al.: “**WS-Resource Framework**”, version 1.0, Globus Alliance, IBM, Computer Associates International, Fujitsu Laboratories of Europe, and Hewlett-Packard, May 2004
<http://www.globus.org/wsrf>
- [123] B. Siddiqui: “**Web Services Security, Part 1**”, March 2003
<http://webservices.xml.com/pub/a/ws/2003/03/04/security.html>
- [124] B. Siddiqui: “**Web Services Security, Part 2**”, April 2003
<http://webservices.xml.com/pub/a/ws/2003/04/01/security.html>
- [125] B. Siddiqui: “**Web Services Security, Part 3**”, May 2003
<http://webservices.xml.com/pub/a/ws/2003/05/13/security.html>
- [126] B. Siddiqui: “**Web Services Security, Part 4**”, July 2003
<http://webservices.xml.com/pub/a/ws/2003/07/22/security.html>
- [127] B. Siddiqui: “**XML Canonicalization**”, September 2002
<http://webservices.xml.com/pub/a/ws/2002/09/18/c14n.html>
- [128] B. Siddiqui: “**XML Canonicalization, Part 2**”, October 2002
<http://webservices.xml.com/pub/a/ws/2002/10/09/canonicalization.html>
- [129] S. K. Tan, Y. Ge, K. S. Tan, C. W. Ang, N. Ghosh: “**Dynamically Loadable Protocol Stacks**”, IEEE Internet Computing, Vol. 8, No. 2, March/April 2004, pp. 19-25
- [130] A. Wool: “**A Quantitative Study of Firewall Configuration Errors**”, Computer, Vol. 37, No. 6, June 2004, pp. 62-67
- [131] S. S. Y. Shim, L. Gong, A. D. Rubin, L. Gwennap: “**Securing the High-Speed Internet**”, Computer, Vol. 37, No. 6, June 2004, pp. 33-35

- [132] D. Geer: “**Jus How Secure Are Security Products**”, Computer, Vol. 37, No. 6, June 2004, pp. 14-16
- [133] R. Friend: “**Making the Gigabit IPsec VPN Architecture Secure**”, Computer, Vol. 37, No. 6, June 2004, pp. 54-60
- [134] F. Raynal, et al.: “**Honeypot Forensics Part I: Analyzing the Network**”, IEEE Security & Privacy, Vol. 2, No. 4, July/August 2004, pp. 72-78
- [135] F. Raynal, et al.: “**Honeypot Forensics Part II: Analyzing the Compromised Host**”, IEEE Security & Privacy, Vol. 2, No. 5, September/October 2004, pp. 77-80
- [136] J. G. Levine, J. B. Grizzard, H. L. Owen: “**Using Honeypots to Protect Large Enterprise Networks**”, IEEE Security & Privacy, Vol. 2, No. 6, November/December 2004, pp. 73-75
- [137] S. L. Garfinkel: “**E-Mail Based Identification and Authentication: An Alternative to PKI**”, IEEE Security & Privacy, Vol. 1, No. 6, November/December 2003, pp. 20-26
- [138] G. Cybenko: “**Security Alchemy**”, IEEE Security & Privacy, Vol. 2, No. 6, November/December 2004, pp. 5-5
- [139] O. S. Saydjari: “**Multilevel Security: Reprise**”, IEEE Security & Privacy, Vol. 2, No. 5, September/October 2004, pp. 64-67
- [140] W. Liu, G. Gu: “**Security Issues in Grid Environment**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 510-514
- [141] I. S. Abdullah, D. A. Menasce: “**A Unified Architecture for the Implementation of Security Protocols**”, Proceedings of the 2003 International Conference of Computer Applications in Industry and Engineering (CAINE03), November 2003, Las Vegas, Nevada, USA
- [142] D. A. Menasce: “**Composing Web Services: A QoS View**”, IEEE Internet Computing, Vol. 8, No. 6, November/December 2004, pp. 88-90
- [143] D. A. Menasce: “**Allocating Applications in Distributed Computing**”, IEEE Internet Computing, Vol. 9, No. 1, January/February 2005, pp. 90-92
- [144] P. Bellavista, A Corradi, C. Stefanelli: “**Application-Level QoS Control for Video-on-Demand**”, IEEE Internet Computing, Vol. 7, No. 6, November/December 2003, pp. 16-24
- [145] Q. Han, S. Gutierrez-Nolasco, N. Venkatasubramanian: “**Reflective Middleware for Integrating Network Monitoring with Adaptive Object Messaging**”, IEEE Network, Vol. 18, No. 1, January/February 2004, pp. 56-65
- [146] V. Pandey, D. Ghosal, B. Mukherjee: “**Exploiting User Profiles to Support Differentiated Services in Next-Generation Wireless Networks**”, IEEE Network, Vol. 18, No. 5, September/October 2004, pp. 40-48
- [147] W. Liu, G. Gu: “**Providing Data Transfer with QoS as Agreement-Based Service**”, Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 2004), September 2004, Shanghai, China, pp. 344-353
- [148] S. Raina: “**Quality of Service to Subscriber**”, IEE Telecommunications Quality of Services: The Business of Success (QoS 2004), March 2004, London, UK, pp. 48-51
- [149] J. Riedl: “**Personalization and Privacy**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 29-31

- [150] M. P. Singh: “**Privacy and Open Services**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 4-5
- [151] D. Farber: “**Balancing Security and Liberty**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 96-96
- [152] J. F. McCarthy: “**The Virtual World Gets Physical**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 48-53
- [153] N. Ramakrishnan, B. J. Keller, B. J. Mirza, A. Y. Grama, G. Karypis: “**Privacy Risks in Recomender Systems**”, IEEE Internet Computing, Vol. 5, No. 6, November/December 2001, pp. 54-62
- [154] J. Waidya, C. Clifton: “**Privacy-Preserving Data Mining: Why, How, and When**”, IEEE Security & Privacy, Vol. 2, No. 6, November/December 2004, pp. 19-27
- [155] J. McCarthy: “**Reap the Benefits of Document Style Web services**”, June 2002
<http://www-128.ibm.com/developerworks/webservices/library/ws-docstyle.html>
- [156] K. J. Hole, E. Dyrnes, P. Thorsheim: “**Securing Wi-Fi Networks**”, Computer, Vol. 38, No. 7, July 2005, pp. 28-34
- [157] P. S. Rosenbloom: “**A New Framework for Computer Science and Engineering**”, Computer, Vol. 37, No. 11, November 2004, pp. 23-28
- [158] S. Vinoski: “**Web Services References**”, IEEE Internet Computing, Vol. 9, No. 2, March/April 2005, pp. 94-96
- [159] S. Vinoski: “**WS-Nonexistent Standards**”, IEEE Internet Computing, Vol. 8, No. 6, November/December 2004, pp. 94-96
- [160] R. E. Filman: “**Postmodern Software Development**”, IEEE Internet Computing, Vol. 9, No. 1, January/February 2005, pp. 4-6
- [161] A. Pashtan, A. Heusser, P. Scheuermann: “**Personal Service Areas for Mobile Web Applications**”, IEEE Internet Computing, Vol. 8, No. 6, November/December 2004, pp. 34-39
- [162] J. O. Kephart, D. M. Chess: “**The Vision of Autonomic Computing**”, Computer, Vol. 36, No. 1, January 2003, pp. 41-50
- [163] W. Provost: “**What Interoperability Isn’t**”, September 2003
<http://webservices.xml.com/pub/a/ws/2003/09/02/interop.html>
- [164] R. Wilson: “**Tech Trends Will Topple Tradition**”, EETimes, January 2005
<http://www.eet.com/showArticle.jhtml?articleID=57300162>
- [165] S. Kapp: “**802.11: Leaving the Wire Behind**”, IEEE Internet Computing, Vol. 6, No. 1, January/February 2002, pp. 82-85
- [166] W. Stallings: “**IEEE 802.11: Wireless LANs from a to n**”, IT Professional, Vol. 6, No. 5, September/October 2004, pp. 32-37
- [167] C. Petzold: “**Programming Microsoft Windows with C#**”, Microsoft Press, 2002
- [168] J. Richter: “**Applied Microsoft .NET Framework Programming**”, Microsoft Press, 2002
- [169] D. J. Reilly: “**Designing Microsoft ASP.NET Applications**”, Microsoft Press, 2002

- [170] M. Powell: “**Programming with Web Services Enhancements 2.0**”, Microsoft Developer Network, May 2004
<http://msdn.microsoft.com/webservices/webservices/building/wse/default.aspx?pull=/library/en-us/dnwse/html/programwse2.asp>
- [171] L. Braginski, M. Powell: “**Take IIS Customization to the Next Level by Writing Filters and Script Interpreters**”, Microsoft Systems Journal, April 1998
- [172] J. Clark, S. DeRose: “**XML Path Language (XPath) Version 1.0**”, W3C Recommendation, November 1999
<http://www.w3.org/TR/xpath>
- [173] P. Hallam-Baker, C. Kaler, R. Monzillo, A. Nadalin: “**Web Services Security X.509 Certificate Token Profile**”, OASIS Standard 200401, March 2004

11 Životopis

Rođen sam 11. srpnja 1979. godine u Čakovcu. Osnovnu školu završio sam u Gornjem Mihaljevcu, a srednju elektrotehničku u Čakovcu. Tijekom srednjoškolskog obrazovanja više sam puta nagradivan od strane Nastavničkog vijeća Tehničke, industrijske i obrtničke škole Čakovec za postignute uspjehe u ovladavanju nastavnim programom te na županijskim i državnim natjecanjima. Na kraju školovanja nagrađen sam naslovom najboljeg maturanta Tehničke, industrijske i obrtničke škole Čakovec. Godine 1998. upisao sam dodiplomski studij na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Znanstveno-nastavno vijeće Fakulteta elektrotehnike i računarstva odobrilo mi je završetak studija s naglaskom na znanstveno-istraživačkom radu. Diplomirao sam u lipnju 2003. godine na studiju računarstva diplomskim radom pod naslovom "Sigurni prijenos podataka u mrežama s posredničkim sustavima". Pisani rad i usmena obrana ocijenjeni su izvrsnom ocjenom. Tijekom studija primio sam priznanje "Josip Lončar" od strane Fakultetskog vijeća Fakulteta elektrotehnike i računarstva kao jedan od najboljih studenata na 4. godini studija.

Tijekom dviju završnih godina studija obnašao sam funkciju demonstratora na Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave. Istovremeno, aktivno sam sudjelovao u provedbi projekta "*Middleware Architecture in New Generation Networks*" u suradnji tvrtke Ericsson Nikola Tesla d.d. i Fakulteta elektrotehnike i računarstva. Tijekom ljeta 2001. i 2002. godine sudjelovao sam u ljetalnim radionicama u tvrtci Ericsson Nikola Tesla d.d. u organizaciji Zavoda za elektroniku, mikroelektroniku, računalne i intelligentne sustave i istoimene tvrtke u ukupnom trajanju od tri mjeseca. Tijekom 2001. godine u suradnji Fakulteta elektrotehnike i računarstva i tvrtke Intesis d.o.o. iz Zagreba sudjelovao sam na projektu izrade jezičnog procesora kao voditelj studentske grupe od 12 članova.

Akademске godine 2003./2004. upisao sam poslijediplomski studij za stjecanje akademskog stupnja magistra znanosti iz područja računarskih znanosti, na smjeru Jezgra računarstva. Od siječnja 2004. godine zaposlen sam na Zavodu za elektroniku, mikroelektroniku, računalne i intelligentne sustave na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Istraživački rad vezan je uz tehnologiski projekt "*Okvirje posrednika javnog informacijskog sustava*" pod pokroviteljstvom Ministarstva znanosti, obrazovanja i športa Republike Hrvatske. Godine 2004. izabran sam u naslovno zvanje mladeg asistenta za područje tehničkih znanosti, polje Računarstvo, grupa predmeta Računarska tehnika. U okviru nastavnih aktivnosti na Zavodu, aktivno sudjelujem u provedbi auditornih i laboratorijskih vježbi iz više predmeta na dodiplomskom studiju računarstva. Istovremeno se

aktivno uključujem u organizaciju i provedbu ljetnih radionica u tvrtci Ericsson Nikola Tesla d.d. 2004. i 2005. godine.

Na međunarodnim konferencijama sudjelovao sam objavljinjem rada iz područja sigurnosti komunikacije u raspodijeljenim posredničkim sustavima. Dva rada iz područja pokretljivih višemedijskih portala i sigurnosti komunikacije u raspodijeljenim posredničkim sustavima objavio sam u zbornicima ljetnih škola tvrtke Ericsson Nikola Tesla d.d. Krajem 2005. godine boravio sam u Sjedinjenim Američkim Državama gdje sam tijekom dvotjednog stručnog posjeta prestižnim američkim sveučilišnim i industrijskim središtima University of California at Berkeley, University of California at Irvine, Santa Clara University i Intel Corporation Inc. održao niz predavanja s temom "PIE - Programmable Internet Environment".

12 Sažetak

Računarstvo zasnovano na uslugama predstavlja novu paradigmu za razvoj i izvođenje raspodijeljenih aplikacija zasnovanih na globalnoj mreži Internet. Raspodijeljene aplikacije zasnovane na uslugama grade se povezivanjem programskih usluga dostupnih u mreži Internet. Programski međusloj za potporu izgradnji i izvođenju raspodijeljenih aplikacija zasnovanih na uslugama pruža funkcionalnosti za postavljanje, povezivanje i komunikaciju programskih usluga te izvođenje raspodijeljenih aplikacija izgrađenih povezivanjem usluga. Dio programskog međusloja računalnih sustava zasnovanih na uslugama je sustav prividne mreže. Prividna mreža uspostavlja logički komunikacijski prostor za sigurnu i pouzdanu komunikaciju programskih usluga. Primjenom sustava prividne mreže, nad zajedničkom komunikacijskom infrastrukturom mreže Internet oblikuju se logički odvojeni komunikacijski prostori prilagođeni zahtjevima pojedinih raspodijeljenih aplikacija. Sustav prividne mreže oblikovan i izgrađen u okviru ovog magistarskog rada pruža potporu prenosivosti raspodijeljenih aplikacija zasnovanih na uslugama, otpornosti aplikacija na kvarove čvorova i komunikacijskih veza fizičke mreže, sigurnosti komunikacije programskih usluga te privatnosti korisnika i pružatelja programskih usluga.

13 Summary

“Virtual Network of Service-Oriented Computing Systems”

Service-oriented computing represents an emerging paradigm for development and execution of distributed Internet-based applications. Development of service-oriented distributed applications is based on a composition of services available on the Internet. Development and execution of such applications is facilitated by an application-level middleware. Application-level middleware supports the deployment, discovery, composition, and communication of individual services, as well as the execution of distributed applications in the form of composite services. An integral part of application-level middleware is a virtual network. Virtual network enables creation of logically isolated communication spaces for a secure and reliable communication of services. Logical communication spaces are established on top of shared Internet infrastructure. Customization of virtual network enables an on-demand adjustment of logical communication space to the requirements of particular distributed application. In this thesis, the design and implementation of virtual network of service-oriented computing systems is presented. Virtual network provides the support for distributed service-oriented application portability, fault-tolerance regarding to failures in network nodes and communication links, secure communication of services, and privacy of their consumers and providers.

14 Ključne riječi

Ključne riječi:

raspodijeljeni računalni sustavi, prividna mreža, arhitektura zasnovana na uslugama, komunikacijski posrednik, *Web Services* radni okvir, Internet, prenosivost raspodijeljenih programskih sustava, otpornost na kvarove, učinkovitost, sigurnost

Keywords:

distributed computing systems, virtual network, service-oriented architecture, communication middleware, *Web Services* framework, Internet, distributed application portability, fault-tolerance, performance, security
