Raspodijeljeni sustavi

Pitanja i odgovori iz ispita, rokova i pitanja za provjeru znanja

Verzija 0.1
6.2.2013.

Sadržaj

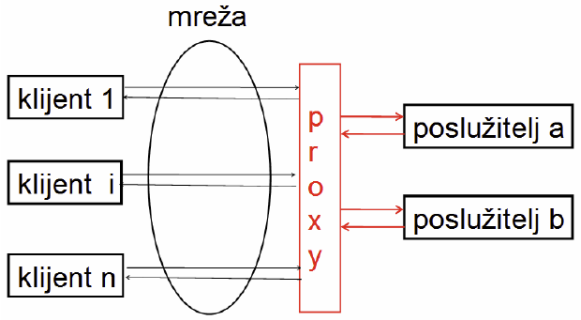
Teorijski zadaci

Prvi ciklus

Objasnite pojam konkurencijske transparentnosti raspodijeljenih sustava.

Za raspodijeljeni sustav kažemo da posjeduje konkurencijsku transparentnost ukoliko on omogućava da više različitih korisnika istodobno (konkurentno) rabi isto sredstvo, a da pri tome oni sami toga nisu niti svjesni.

Skicirajte i objasnite ulogu zastupnika poslužitelja (*proxy*). Ukoliko pretpostavimo da zastupnik poslužitelja obavlja zadaću priručne pohrane (*cache*), što će se dogoditi pri ponovnom zahtjevu za istim podatkom?



Zastupnik poslužitelja (*proxy*) posreduje između klijenata i poslužitelja tako da od klijenata prikriva broj i lokaciju poslužitelja te način na koji su povezani (tj. omogućava replikacijsku transparentnost). Ukoliko je vrijeme između ta dva zahtjeva za istim podatkom relativno kratko, zastupnik poslužitelja će imati kopiju tog podatka pohranjenu u priručnom spremištu te će ju isporučiti klijentu bez ponovnog kontaktiranja poslužitelja.

Objasnite razliku između perzistentne i tranzijentne komunikacije procesa? Navedite po jedan primjer za obje vrste komunikacije.

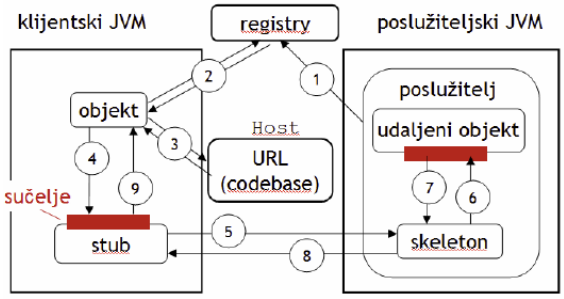
Kod **perzistentne** komunikacije procesa jamči se isporuka poruke ukoliko pošiljatelj i primatelj nisu istodobno dostupni i to tako da se poruka pohranjuje u sustavu i isporučuje primatelju kada to bude moguće. Kod **tranzijentne** komunikacije ne jamči se isporuka poruke ukoliko pošiljatelj i primatelj nisu istodobno dostupni.

Primjeri za perzistentnu komunikaciju su **objavi-pretplati** i **komunikacija porukama**. Primjeri za tranzijentnu komunikaciju su socket **TCP** i **UDP**.

U tablici su prikazane aktivnosti na poslužiteljskoj strani kod socketa TCP. Da bi redoslijed postao ispravan, popunite tablicu odgovarajućim rednim brojevima aktivnosti.

1. socket()
2. bind()
3. listen()
4. accept()
5. read()
6. write()
7. close()

Skicirajte model pozivanja udaljene metode Java RMI (Remote Method Invocation), uz pretpostavku da se klasa stub učitava dinamički. Navedite redoslijed koraka u komunikaciji koji je potreban da bi klijent pozvao metodu dostupnu na poslužitelju.



Koraci u komunikaciji su sljedeći:

1. Poslužitelj definira codebase udaljenog objekta i registrira ga pod odabranim imenom.
2. Klijent (ili objekt na klijentu) od registrya traži i dobiva referencu na udaljeni objekt koristeći
3. registrirano ime.
4. Klijent traži i dobiva klasu stub koristeći codebase.
5. Klijent poziva metodu stuba dostupnu na klijentskom računalu.
6. Stub serijalizira parametre i šalje ih skeletonu.
7. Skeleton deserijalizira parametre i poziva metodu udaljenog objekta.
8. Udaljeni objekt vraća rezultat izvođenja metode skeletonu.
9. Skeleton serijalizira rezultat i šalje ga stubu.
10. Stub deserijalizira rezultat i dostavlja ga klijentu.

Objasnite sličnosti i razlike u obilježjima komunikacije između dva komunikacijska modela podržana s JMS (Java Messaging Service)?

JMS podržava komunikaciju porukama i model objavi-pretplati. Obje vrste komunikacije su vremenski neovisne zato što pošiljatelj i primatelj ne moraju istovremeno biti dostupni da bi se komunikacija mogla ostvariti. Kod komunikacije porukama pošiljatelj mora znati identifikator odredišta, dok je kod modela objavi-pretplati komunikacija anonimna. Komunikacija je perzistentna i asinkrona u oba slučaja. Komunikacija se pokreće na načelu pull kod komunikacije porukama, a na načelu push kod modela objavi-pretplati.

Usporedite grozd i splet računala s obzirom na kategorije i značajke koje su prikazane u tablici. Za svaku od ponuđenih kategorija, u stupac „Splet ili grozd“ upišite splet ukoliko ta značajka karakterizira splet računala ili grozd ukoliko ona karakterizira grozd računala.

Kategorija	Značajka	Splet ili grozd
Aplikacije	Izvođenje računalno zahtjevnih aplikacija	Grozd
	Dijeljenje raznovrsnih sredstava u globalnoj mreži	Splet
Tehnologija	Primjena vlasničkih i standardiziranih tehnologija	Grozd
	Primjena standardiziranih tehnologija	Splet
Geografska raspodijeljenost	Elementi sustava globalno raspodijeljeni	Splet
	Elementi sustava na bliskoj geografskoj udaljenosti	Grozd
Upravljanje	Središnje upravljanje sredstvima sustava	Grozd
	Više administrativnih domena za upravljanje sustavom	Splet
Povezanost	Labavo povezane strukture	Splet
	Čvrsto povezane strukture	Grozd
Prilagodljivost	Zatvorena okolina	Grozd
	Otvorena okolina	Splet

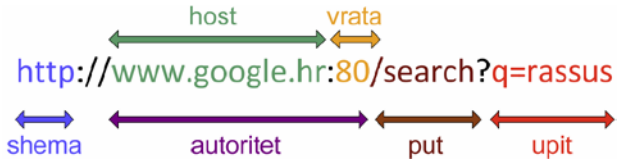
Kako se izvodi pretraživanje kod strukturiranih, a kako kod nestrukturiranih sustava sustava P2P (peer-to-peer)? Koji od ovih sustava su skalabilni i zašto?

U nestrukturiranim sustavima P2P, podatak je pohranjen na peeru koje ga kreira, a njegova kopija može biti pohranjena i na nekim drugim peerovima u mreži. Zbog toga se u ovim sustavima pretraživanje izvodi preplavljivanjem mreže i slučajnim izborom (random walk). Kod strukturiranih sustava P2P, podatak je pohranjen na peeru koji je zadužen za ključ tog podatka. Pretraživanje se provodi traženjem podatka po njegovom ključu. Strukturirani sustavi su skalabilni zato što se kod njih pretraživanje odvija u log(n) koraka, gdje je n broj peerova u mreži.

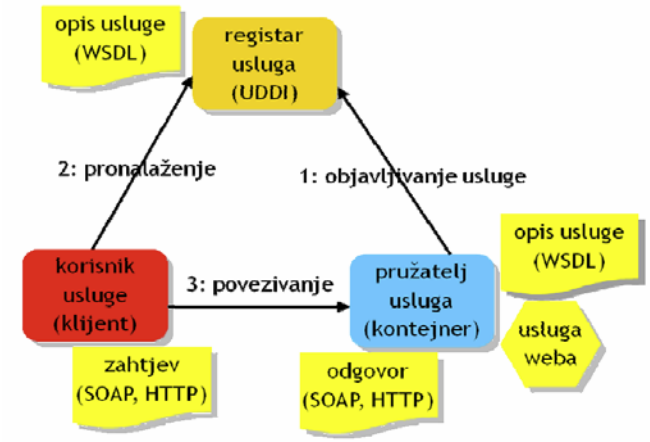
Objasnite razliku između programskog agenta i pokretnog programskog agenta. Što je to agentska platforma i koje tri vrste funkcija ona sadrži?

Za razliku od programskog agenta koji je statičan, pokretni programski agent može samostalno migrirati u mreži. Agentska platforma je osnovna programska oprema koja agentu omogućuje pokretljivost. Ona se sastoji od sljedeća tri sloja funkcija: agentske, sigurnosne i komunikacijske funkcije.

Označite karakteristične dijelove URI-ja na sljedećem primjeru.



Skicirajte arhitekturu i slijed operacija za korištenje usluge weba. Ukratko objasnite redoslijed operacija pri korištenju usluga weba.



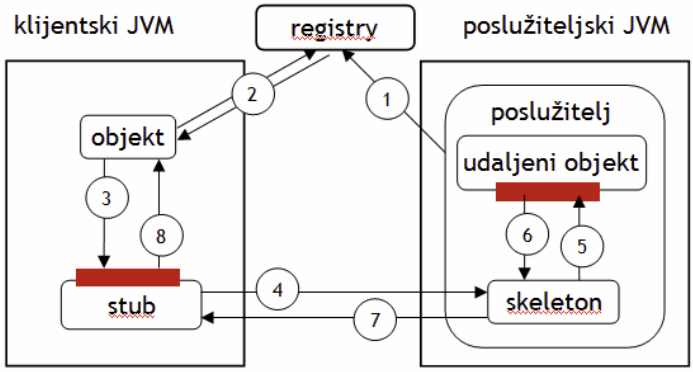
Redoslijed operacija pri korištenju usluge weba je sljedeći:

1. **Objavljivanje** usluge: pružatelj usluge vrši registraciju usluge u Registru usluga koristeći WSDL opis usluge.
2. **Pronalaženje** usluge: korisnik usluge (klijent) pronalazi odgovarajuću uslugu z registru usluga.
3. **Povezivanje**: temeljem specifikacije u WSDL opisu usluge, klijent šalje zahtjev za uslugom, te od usluge dobiva odgovor. Ovisno o usluzi, zahtjev i odgovor mogu biti upućeni korištenjem protokola SOAP ili HTTP.

Usporedite model klijent-poslužitelj s uslugom weba prema značajkama koje su navedene u sljedećoj tablici. Za svaku od ponuđenih kategorija, u stupac „KP ili WS“ upišite KP ukoliko ta značajka karakterizira model klijent-poslužitelj ili WS ukoliko ona karakterizira uslugu weba.

Kategorija	Značajka	KP ili WS
Primjena	Unutar tvrtke	KP
	Između tvrtki	WS
Programski jezici	Neovisnost o programskom jeziku	WS
	Ograničen skup programskih jezika	KP
Komunikacija između komponenti	Proceduralno	KP
	Razmjenom poruka	WS
Transportni mehanizmi	Različiti transportni mehanizmi	WS
	Jedan transportni mehanizam	KP
Povezanost	Slabo povezane strukture	WS
	Čvrsto povezane strukture	KP
Učinkovitost procesiranja	Velika	KP
	Manja	WS

Skicirajte model pozivanja udaljene metode Java RMI (Remote Method Invocation). Navedite korake u komunikaciji potrebne da bi klijent pozvao metodu dostupnu na poslužitelju, uz pretpostavku da je klasa stub već instalirana na klijentskoj strani.



1. Poslužitelj registrira udaljeni objekt pod odabranim imenom.
2. Klijent od *registrya* traži referencu na udaljeni objekt koristeći registrirano ime.
3. Klijent poziva metodu *stuba* dostupnu na klijentskom računalu.
4. *Stub* serijalizira parametre i šalje ih *skeletonu*.
5. *Skeleton* deserijalizira parametre i poziva metodu udaljenog objekta.
6. Udaljeni objekt vraća rezultat izvođenja metode *skeletonu*.
7. *Skeleton* serijalizira rezultat i šalje ga *stubu*.
8. *Stub* deserijalizira rezultat i dostavlja ga klijentu.

Navedite vrste funkcija koje sadrži agentska platforma.

***

Čemu služi protokol SOAP?

*** odreduje format poruke, komunikaciju web usluga

Navedite i ukratko objasnite vrste usluga weba.

***
REST
-http , GET,POST,DELETE
RPC
-prijenos pohranjenih procedura
-WSDL,SOAP
PORUKE
XML, WSDL,SOAP

Skicirajte i objasnite primjer komunikacije porukama između dva procesa/objekta (primatelja i pošiljatelja). Kakva je komunikacija porukama s obzirom na vremensku ovisnost primatelja i pošiljatelja?

U komunikaciji između pošiljatelja i primatelja rep sudjeluje kao posrednik. Pošiljatelju se u načelu garantira isporuka poruke u primateljev rep, ali ne i isporuka poruke primatelju. Primatelj može pročitati poruku iz repa u bilo kojem budućem trenutku. Stoga su pošiljatelj i primatelj poruke vremenski neovisni.

U tablicama su prikazane metode na klijentskoj i poslužiteljskoj strani socketa TCP. Upišite ispravan redoslijed izvođenja metoda u tablice.

Klijent	Poslužitelj
socket()	1. socket()
connect()	2. bind()
write()	3. listen()
read()	4. accept()
close()	5. read()
*	6. write()
*	7. close()

Objasnite pojam migracijske transparentnosti raspodijeljenih sustava na primjeru raspodijeljenog sustava weba.

***

Skicirajte i objasnite razliku između komunikacije procesa temeljene na načelu *pull* i *push*.

***

Objasnite razliku u obilježjima komunikacije između modela objavi-pretplati i komunikacije porukama?

***

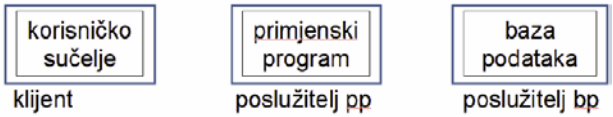
Objasnite pojam skalabilnosti raspodijeljenog sustava.

Raspodijeljeni sustav je skalabilan ukoliko posjeduje sposobnost prilagodbe povećanom broju korisnika i sredstava, njihovoj rasprostranjenosti te načinu upravljanja sustavom.

Objasnite pojam migracijske transparentnosti raspodijeljenog sustava.

Za raspodijeljeni sustav kažemo da posjeduje migracijsku transparentnost ukoliko on prikriva promjenu lokacije nekog sredstva na način da ta promjena ne utječe na način pristupa tom sredstvu.

Skicirajte trorednu arhitekturu klijent-poslužitelj te na proizvoljnom primjeru aplikacije objasnite ulogu svake razine u cjelokupnoj arhitekturi.



Primjer su aplikacije weba, gdje klijentski program koji se izvodi na klijentskom računalu nikada ne pristupa direktno bazi podataka, već posredno preko aplikacije weba. Klijentski program prikazuje korisničko sučelje i komunicira s aplikacijom weba koja obavlja cjelokupnu logiku usluge i pristupa potrebnim podacima.

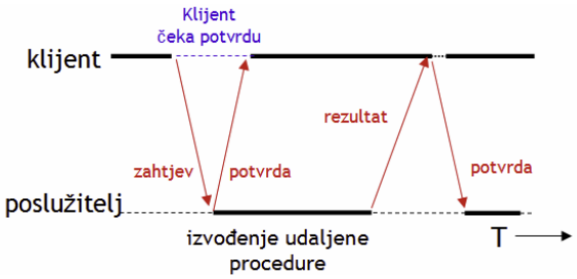
Objasnite razliku između sinkrone i asinkrone komunikacije.

Dok je kod sinkrone komunikacije pošiljatelj blokiran nakon slanja poruke sve do primitka potvrde o isporuci, kod asinkrone komunikacije pošiljatelj nije blokiran te nastavlja procesiranje odmah nakon slanja.

Navedite obilježja komunikacije socketom UDP.

Ova komunikacija se temelji na modelu klijent-poslužitelj, gdje oba moraju istovremeno biti aktivna da bi se komunikacija mogla ostvariti. Komunikacije je tranzijentna i asinkrona, a može se koristiti za implementaciju komunikacije na načelu *pull* ili *push*.

Skicirajte tijek komunikacije između klijenta i poslužitelja te objasnite odgođeni sinkroni poziv udaljene procedure RPC (Remote Procedure Call).



Kod odgođenog sinkronog poziva udaljene procedure, klijent nije blokiran dok čeka rezultat izvođenja, već nastavlja s radom nakon uspješnog primitka potvrde. Kasnije mu poslužitelj šalje rezultat koristeći drugi asinkroni poziv udaljene procedure.

Objasnite opći format poruka protokola HTTP. Navedite kako glasi potpun i apsolutan URI koji identificira resurs zatražen u zahtjevu, ako prva 2 retka HTTP zahtjeva sadrže sljedeće podatke:
GET /predmet/rassus HTTP/1.1
Host: www.fer.hr

Opći format poruka protokola HTTP sastoji se od početnog retka, polja zaglavlja te tijela poruke. Potpun i apsolutan URI je http://www.fer.hr/predmet/rassus .

Korisnik nakon ispunjavanja obrasca na Web-u odabire opciju Submit, čime pošalje podatke Web-poslužitelju na adresu *www.tel.fer.hr/obrazac/accept* korištenjem protokola HTTP verzije 1.1. Kojim se HTTP zahtjevom šalju podaci poslužitelju i kako je definiran prvi redak zahtjeva?

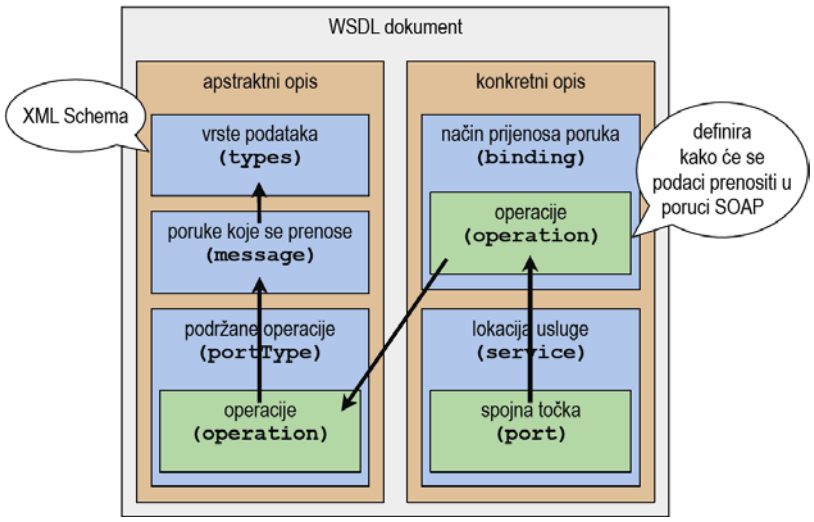
Podaci se šalju zahtjevom POST.

Prvi redak je definiran na sljedeći način:
POST /obrazac/accept HTTP 1.1 .

Navedite dva osnovna načina rada protokola SOAP i objasnite kako se poruka SOAP šalje pomoću protokola HTTP.

Dva osnovna načina rada koje podržava protokol SOAP su poziv udaljene procedure te razmjena dokumenata i poruka. Poruka SOAP, koja je pisana jezikom XML, se sastoji od zaglavlja i tijela. Prilikom slanja poruke SOAP protokolom HTTP, i zaglavlje i tijelo poruke SOAP se nalaze u tijelu poruke HTTP.

Objasnite i skicirajte sadržaj apstraktnog i konkretnog opisa u strukturi dokumenta WSDL.



Apstraktni opis:

1. *types*: definira vrste podataka neovisne o platformi i jeziku (koristi se XML Schema),
2. *message*: definiraju ulazne i izlazne poruke koje se mogu koristiti kao parametri usluge,
3. *operation*: predstavlja jednu operaciju/metodu/proceduru koja je definirana u usluzi, a sastoji se od definicija ulaznih, izlaznih i iznimnih poruka koje se mogu razmjenjivati korištenjem ove operacije i
4. *portType*: koristi poruke (pod 2) da bi opisao sve operacije koje pruža usluga.

Konkretni opis:

1. *binding*: definira kako je konkretna implementacija povezana s operacijama u apstraktnom opisu i definira format u kojem će se poruke prenositi (protokol i elemente) i
2. *service*: definira URI gdje je usluga isporučena tj. na kojoj adresi se može pozvati usluga (taj URI je definiran u spojnoj točci).

Objasnite svojstvo slabe povezanosti usluga kod uslužno orijentirane arhitekture.

Svojstvo slabe povezanosti usluga kod uslužno orijentirane arhitekture (SOA – Service Oriented Architecture) odnosi se na dizajn programske izvedbe usluga. U sustavu SOA, usluge trebaju biti izvedene tako da promjena u jednoj usluzi ne zahtijeva promjenu neke druge usluge. Pri tome, svaka usluga može i dalje nesmetano koristiti neku drugu uslugu. Npr. algoritam koji se koristi u usluzi se može promijeniti bez znanja drugih usluga i druge usluge ju mogu nesmetano koristiti. Bitno je da se sučelja opisana WSDL-om ne promijene.

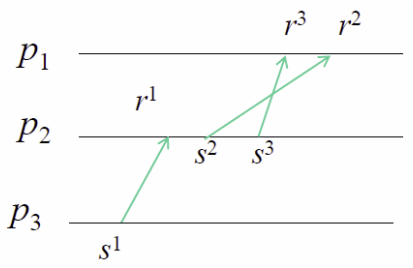
Objasnite za koje je od sljedeća tri svojstva raspodijeljenih sustava značajna komunikacijska složenost algoritama: a) replikacijska transparentnost b) skalabilnost c) otvorenost.

Komunikacijska složenost algoritama je važna za **skalabilnost** raspodijeljenog sustava jer na temelju komunikacijske složenosti možemo zaključiti kako raste generirani promet raspodijeljenog sustava s rastom tog sustava. **Primjer**: komunikacija grupe procesa.

Objasnite model komunikacijskog kanala koji se temelji na uzročnoj slijednosti.

Uzročna slijednost (*causal ordering*) osigurava da uzročno povezani događaji slanja dviju poruka istom primatelju rezultiraju primanjem u slijedu kojim su poslani.

Objasnite zašto za sljedeći primjer vrijedi CO ili vrijedi non-CO?



Za primjer vrijedi non-CO zato što proces p_1 prima poruke od p_2 drugačijim redoslijedom od redoslijeda slanja, a pri tome je slanje poruke 3 uvjetovano slanjem poruke 2.

Navedite i objasnite operacije koje implementira programska infrastruktura dijeljenog podatkovnog prostora.

- **write(t)** – dodaj tuple t u raspodijeljeni podatkovni prostor
- **read (s) -> t** – vraća tuple t koji odgovara predlošku s
- **take (s) -> t** – vraća tuple t koji odgovara predlošku s i briše ga iz podatkovnog prostora

Pretpostavite da se sjedište weba sastoji od 2 poslužitelja priključena na Internet preko posrednika (*proxy*). Navedite i objasnite svojstva ovog raspodijeljenog sustava.

Ovaj raspodijeljeni sustav karakteriziraju:

1) replikacijska transparentnost – zato što korisnik nije svjestan koji poslužitelj ga je zapravo poslužio,
2) otpornost na kvarove – jer se kvar jednog poslužitelja može prikriti od korisnika i
3) skalabilnost – zato što ovaj sustav posjeduje sposobnost prilagodbe povedanom broju korisnika.

Ovaj sustav podržava i lokacijsku i migracijsku transparentnost (putem sustava DNS).

Objasnite razliku između web-aplikacija temeljenih na CGI (Common Gateway Interface) i poslužiteljskim skriptama.

CGI (*Common Gateway Interface*) je jednostavno sučelje za pokretanje eksternih programa iz web-poslužitelja na platformski i programski neovisan način. Kod svakog zahtjeva se pokreće novi proces, a podaci između poslužitelja i procesa šalju se preko varijabli okoline i tokova podataka. Nakon svake obrade proces se gasi. Nedostatak CGI-a je što se kod svakog zahtjeva pokreće novi proces i nakon obrade gasi što je zahtjevno za resurse (procesorsko vrijeme i memorija) pa kod velikog broja zahtjeva na poslužitelju to znatno utječe na performanse.

Poslužiteljske skripte (engl. *server side scripts*), dinamički generiraju HTML-dokumente poput CGI-ja, ali je razlika u tome što se za svaki zahtjev ne pokreće novi proces i na taj način se štede resursi.

***** Objasnite tri načina rada posredničkog (*proxy*) poslužitelja**

Proxy poslužitelj posreduje između klijenta i poslužitelja. Svrha – filtriranje informacija, privremeno spremište, anonimiziranje klijenta, bilježenje korištenja mreže, pregledavanje sadržaja.

- **Forward proxy** (lokalna mreža klijenta) – smanjenje korištenja mreže, filtriranje i kontrola sadržaja
- **Open proxy** (javna mreža) – anonimiziranje klijenta
- **Reverse proxy** (lokalna mreža poslužitelja) – enkripcija, uravnoteženje opterećenja, sigurnosne postavke

***** Objasnite svojstva sigurnosti, idempodentnosti i *cacheable* za metode protokola HTTP i označite ih u tablici.**

- Sigurna (safe) – bez posljedica za podatke
- Indempodentna – može se izvršavati više puta
- Cacheable – odgovor se može privremeno spremiti

	sigurnost	*idempodentnost*	*cacheable*
GET	DA	DA	DA
PUT		DA	
DELETE		DA	
HEAD	DA	DA	
POST			

Drugi ciklus

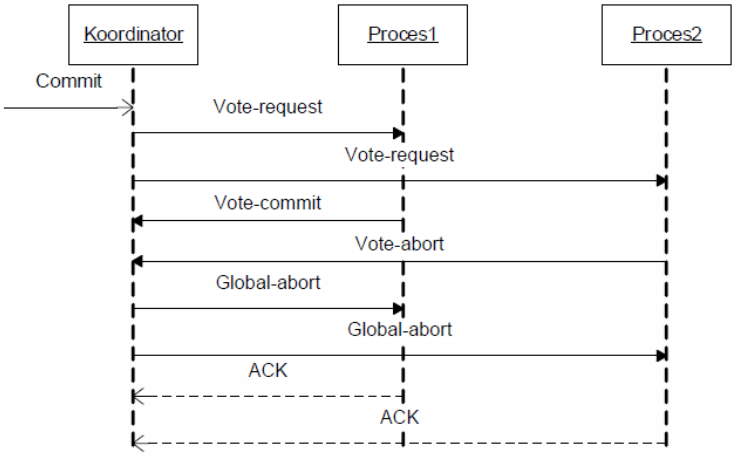
Koje su vrste osnovnih sredstava sadržane u spletu računala? Navedite barem jednu uslugu potrebnu za dijeljene sredstava u spletu računala.

Vrste osnovnih sredstava u spletu računala su:

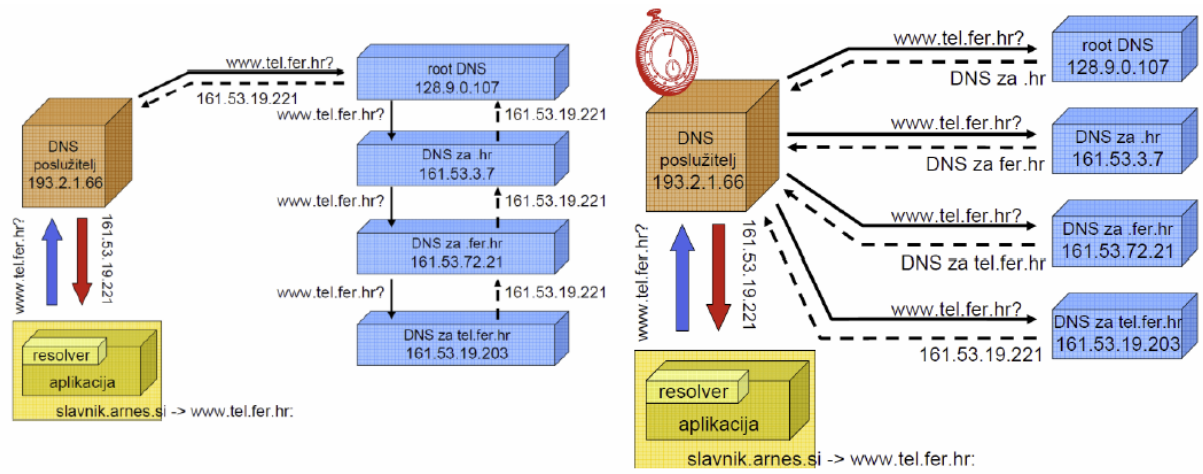
- računalna sredstva (computational resources),
- spremnički prostor (storage systems),
- katalozi sredstava (resource catalogues),
- mrežna sredstva (network resources) i
- osjetila i aktuatori (sensors and actuators).

Jedna od usluga za dijeljenje sredstava je imenička usluga.

Slika prikazuje koordinatora i dva procesa koji koriste protokol two-phase commit za raspodijeljeno izvršavanje operacije commit koju prima koordinator. Pretpostavite da proces 1 prihvaća izvršavanje navedene operacije dok ju proces 2 odbija izvršiti te da nema gubitaka u komunikaciji pri prijenosu poruka. Nadopunite slijedni dijagram prikazan slikom.



Nadopunite skicu te objasnite razliku između rekurzivnog i iterativnog načina rada DNS poslužitelja. Pretpostavite da je u oba slučaja priručno spremište (cache) prazno.

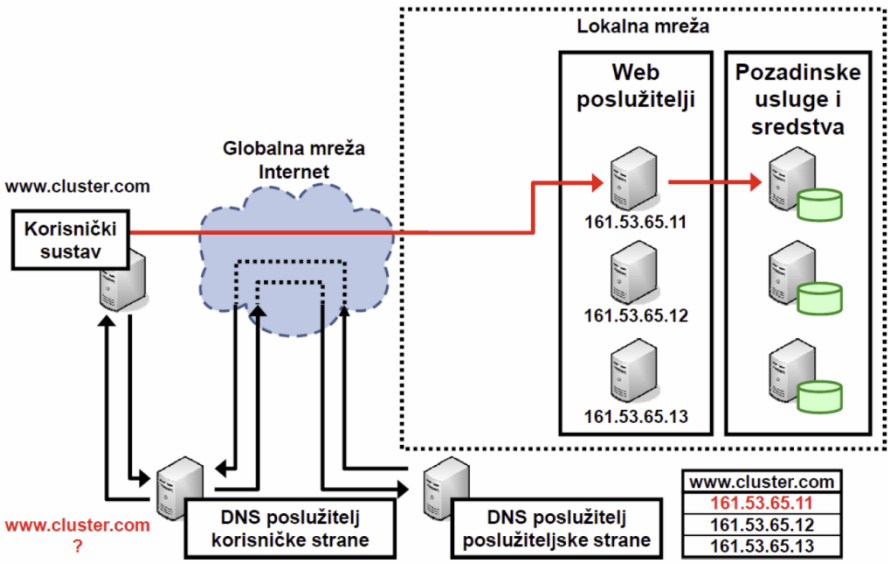


Kod **rekurzivnog načina rada**, poslužitelj vrada ili grešku ili odgovor, ali nikad ne vrada pokazivače na druge DNS poslužitelje.
Kod i**terativnog načina rada** poslužitelj odgovara samo na osnovu informacije koju posjeduje. To znači da vrada grešku, odgovor ili pokazivač na drugi, „bliži“ poslužitelj.

Skicirajte i objasnite slojevitu arhitekturu spleta računala.

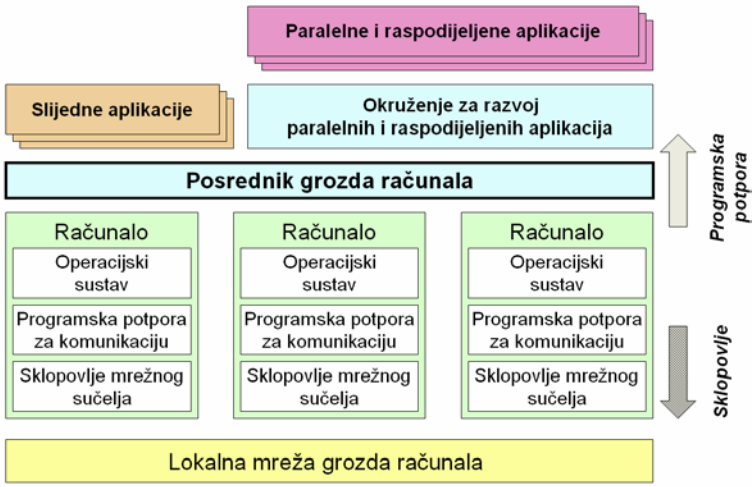
Sloj osnovnih sredstava upravlja osnovnim funkcionalnostima spleta računala. On implementira lokalne operacije koje su specifične svakom sredstvu po vrsti i implementaciji te omogućuje korištenje tih operacija na višim slojevima.
Sloj komunikacijskih protokola definira protokole komunikacije i atutentikacije koji su potrebni za transakcije u spletu.
Sloj protokola za sredstva omogućuje korisniku interakciju s udaljenim resursima i uslugama. On definira protokole za sigurno pregovaranje, pokretanje, praćenje, kontrolu, obračun (engl. accounting) i naplatu dijeljenih operacija i individualnih resursa.
Sloj zajedničkih usluga definira protokole i usluge koji su zaduženi za upravljanje grupom sredstava, a ne za pojedino sredstvo.
Na vrhu se nalazi **sloj korisničkih aplikacija**.

Nadopunite skicu i objasnite primjer ostvarivanja razmjernog rasta sustava primjenom metode <u>DNS poslužitelja</u>.



Kod ostvarivanja razmjernog rasta sustava primjenom metode DNS poslužitelja, korisnički sustav upućuje zahtjev korisničkom DNS poslužitelju. On taj zahtjev prosljeđuje poslužiteljskom DNS poslužitelju. Poslužiteljski DNS poslužitelj odabire jedno od postojećih odredišnih računala te njegovu IP adresu šalje kao odgovor. Korisnički sustav prima odgovor s adresom odredišnog računala i upućuje zahtjev dodijeljenom poslužitelju.

Grozdovi računala zasnovani su na višeslojnoj organizaciji. Skicirajte slojevitu strukturu grozda računala i objasnite slojeve koji se izvode programski.



Sve računala u grozdu su međusobno neovisna, a svako posjeduje operacijski sustav s programskom potporom za komunikaciju. Za izvođenje aplikacija, osim samih računala, potreban je i posrednički sustav grozda računala čiju osnovu čine podsustavi za upravljanje poslovima, za nadgledanje rada i dijeljenje spremničkog prostora te razvojno okruženje. Ukoliko su aplikacije koje se izvode paralelne i raspodijeljene, potrebno je koristiti i odgovarajuće okružje za razvoj takvih aplikacija.

Objasnite što je replika podatka, a što je nekonzistentnost replike podatka.

Replika podatka je jedna kopija podatkovnog objekta u raspodijeljenoj okolini. Nekonzistentnost replika podataka se javlja kada dvije ili više replika u raspodijeljenoj okolini u nekom trenutku u vremenu se nalaze u različitim stanjima.

Navedite i opišite značajke tri osnovna razreda replika podataka u raspodijeljenim sustavima.

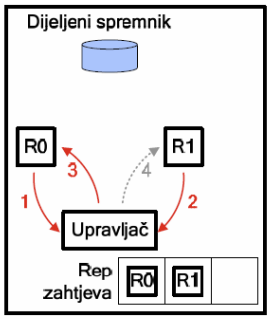
U raspodijeljenim sustavima koriste se sljedeća tri razreda replika:

1. Trajne replike: Osnovni primjerci podataka koji su trajno postavljeni na poslužiteljima. Njihove postavke su statičke i upravljane od strane vlasnika podataka. Nad njima se najčešće ostvaruju operacije čitana podataka i poslužuju primjenom grozdova poslužitelja i replika poslužitelja.

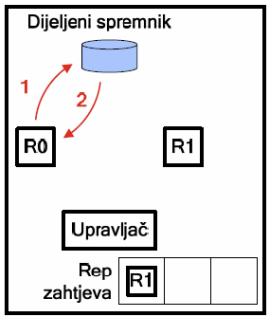
2. Poslužiteljske replike: Poslužitelji podataka stvaraju nove replike trajnih replika i raspoređuju ih na dostupne poslužitelje u mreži. Odabir i raspoređivanje replika ostvaruje se u stvarnom vremenu tijekom rada poslužitelja.

3. Korisničke replike: Postupak repliciranja iniciran je odstrane korisnika putem korisničkih programa. Korisnički programi u lokalne spremnike spremaju pribavljene podatke. Potrebno je održavati konzistentnost lokalnog spremnika s poslužitelje od kojeg su podatci dohvaćeni.

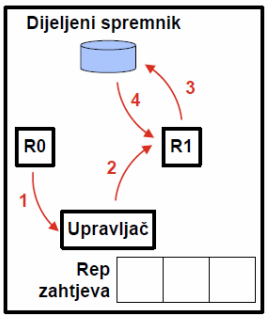
Opišite postupak međusobnog isključivanja dvaju procesa (R0 i R1) primjenom središnjeg upravljača s repom čekanja tako da nacrtate redoslijed operacija i objasnite ih. Nakon zauzimanja dijeljenog spremnika, proces provodi jednu operaciju čitanja ili pisanja nad dijeljenim spremnikom.



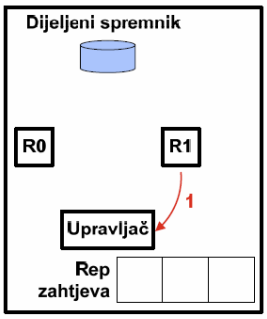
1 – R0 šalje zahtjev za zauzimanje sredstva, zahtjev se sprema u rep
2 – R1 šalje zahtjev za zauzimanje sredstva, zahtjev se stavlja u rep
3 – Kako je zahtjev od R0 stigao prije, upravljač šalje potvrdu R0 i uklanja njegov zahtjev iz repa



1 – R0 provodi operaciju pisanja
2 – R0 prima potvrdu

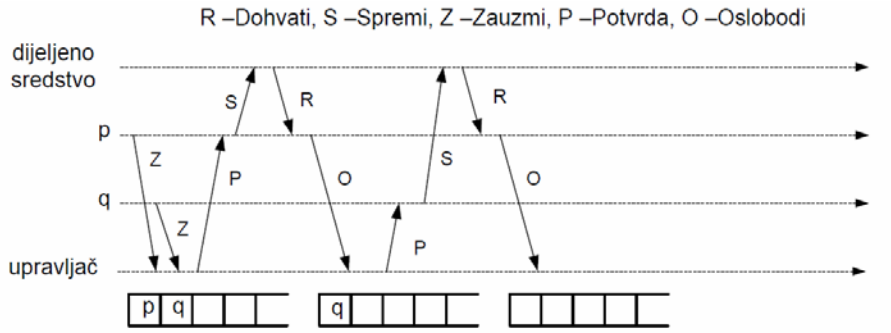


1 – R0 šalje poruku Upravljaču i otpušta pristup
2 – Upravljač šalje poruku dojave R1 te mu dodjeljuje pristup dijeljenom spremniku. Iz repa zahtjeva uklanja se zahtjev od R1
3 – R1 provodi operaciju pisanja
4 – R1 prima potvrdu



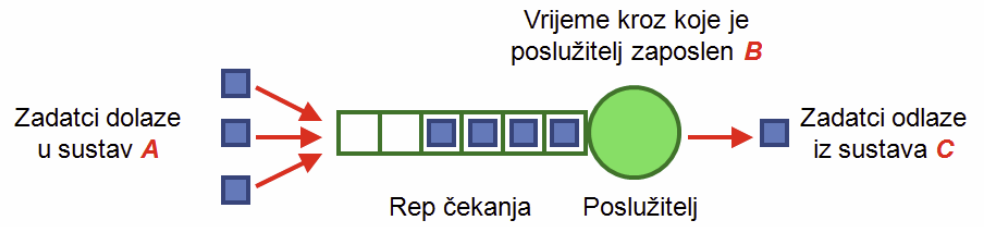
1 – R1 šalje poruku Upravljaču i otpušta pristup

Noviji dijagram (R0=p, R1=q):



- Proces p šalje zahtjev za zauzimanje sredstva, zahtjev se sprema u rep
- Proces q šalje zahtjev za zauzimanje sredstva, zahtjev se stavlja u rep
- Kako je zahtjev od R0 stigao prije, upravljač šalje potvrdu R0 i uklanja njegov zahtjev iz repa
- Proces p provodi operaciju pisanja
- Proces p prima potvrdu
- Proces p šalje poruku Upravljaču i otpušta pristup
- Upravljač šalje poruku dojave procesu q te mu dodjeljuje pristup dijeljenom spremniku. Iz repa zahtjeva uklanja se zahtjev od procesa p
- Proces q provodi operaciju pisanja
- Proces q prima potvrdu
- Proces q šalje poruku Upravljaču i otpušta pristup

Prikažite elemente osnovnog modela repa čekanja. Koje su osnovne veličine, a koje izvedene u modelu repa čekanja? Kako je definirano stacionarno stanje sustava?



Osnovne veličine modela su vrijeme promatranja (T), broj dolazaka (A), broj odlazaka (C) i vrijeme zaposlenosti poslužitelja (B).

Izvedene veličine modela su ulazni ritam (L=A/T), izlazni ritam (X=C/T), srednje vrijeme posluživanja (S=B/C) i zaposlenost poslužitelja (U=B/T).

U stacionarnom stanju sustava je X = L.

Objasnite razliku između ispada sustava i neispravnosti u sustavu.

Ispad sustava je stanje sustava koje se detektira kroz nemogućnost korištenja jedne ili više njegovih usluga. Posljedica je neispravnosti te ukazuje na nju. Neispravnost je nedostatak u programskom kodu, oblikovanju sustava ili komunikacijskom kanalu koji može uzrokovati ispad sustava.

Pretpostavite da se u grupi procesa s ispadima poštuje načelo virtualne sinkronosti. Proces p šalje poruku m grupi procesa G. Tijekom isporuke poruke m dogodi se ispad procesa p. Što se događa isporukom poruke m?

Isporuka se prekida, procesi ignoriraju primljenu poruku.

Pretpostavite da procesi P1 i P2 šalju poruke koje se isporučuju procesima P3 i P4 prema tablici. Navedite koju vrstu pouzdane komunikacije podržava grupa procesa P1, P2, P3 i P4?

Proces P1	Proces P2	Proces P3	Proces P4
šalje m11	šalje m21	prima m11	prima m11
šalje m12	šalje m22	prima m21	prima m12
šalje m13		prima m22	prima m21
		prima m12	prima m13
		prima m13	prima m22

Ova grupa procesa podržava **pouzdani FIFO multicast**.

Navedite obilježja pouzdane komunikacije grupe procesa pod nazivom atomic multicast.

Atomic multicast je pouzdani virtualno sinkroni multicast s potpuno uređenim slijedom poruka.

Objasnite razliku protokola *three-phase commit* u odnosu na *two-phase commit*.

Three-phase commit je sličan protokolu *two-phase commit* osim što koordinator nakon odluke za izvođenje operacije šalje poruku PREPARE_COMMIT na koju procesi odgovaraju s READY_COMMIT. Nakon što primi poruku READY_COMMIT od svih procesa, koordinator šalje GLOBAL_COMMIT.

U spletu računala se koriste 3 komunikacijska sloja: primjenski sloj, sloj prividne mreže i transportni sloj. Ukratko opišite primjenski sloj.

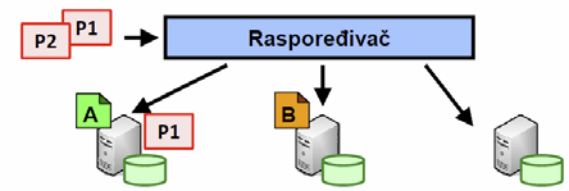
Primjenski sloj je sloj za upravljanje sredstvima u sletu računala koji ostvaruje funkcionalnosti za upravljanje prijenosom velike količine podataka, korištenje udaljenih sredstava na način kao da su lokalno dostupna te izgradnju imenika sredstava u prividnoj mreži spleta računala.

Navedite i opišite osnovne elemente za uspostavu sigurnosti u spletovima računala.

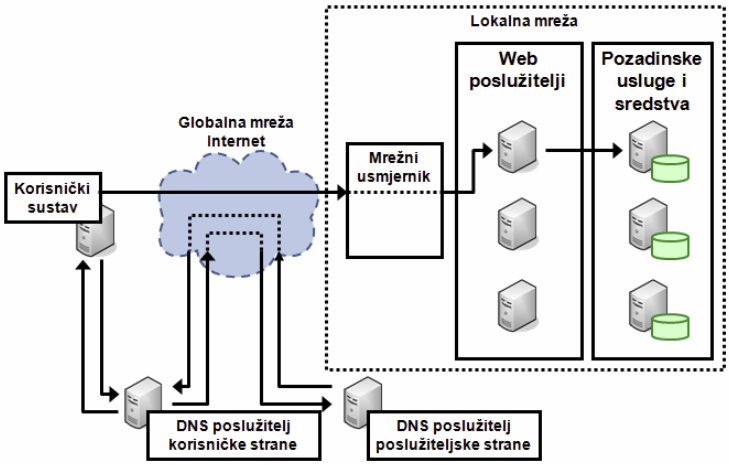
Autentikacija je provjera identiteta korisnika i sredstava u spletu računala. **Autorizacija** je provjera i uspostava kontrole pristupa sredstvima u sustavu spleta računala. Enkripcija podataka je primjena strukture i sadržaja podataka s ciljem zaštite informacija koje se razmjenjuju između sudionika komunikacije.

Na primjeru opišite značajke raspoređivanja zasnovanog na korištenju prostorne lokalnosti.

Kod prostorne lokalnosti, poslovi se raspoređuju na čvorove koji sadrže podatke potrebne za izvođenje posla. Drugim riječima, poslovi se približavaju podacima.

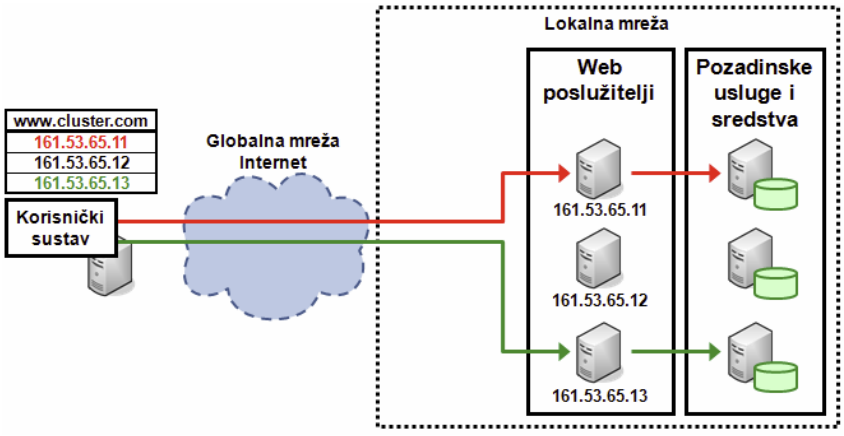


Prikažite i opišite elemente modela grozda računala.



- Korisnički sustav je aplikacija kojom korisnik ostvaruje pristup i koristi sredstva i usluge na grozdu računala.
- DNS poslužitelj korisničke strane je poslužitelj pomoću kojeg korisnički sustav razlučuje adrese udaljenih računala na Internetu.
- DNS poslužitelj poslužiteljske strane je poslužitelj koji razlučuje adrese poslužitelja u lokalnoj mreži.
- Mrežni usmjernik je uređaj koji prihvaća, analizira i usmjerava pristigle zahtjeve.
- Web poslužitelji i pozadinska sredstva i usluge su osnovni elementi grozda računala.

Prikažite primjer ostvarivanja razmjernog rasta sustava primjenom metode *prosljeđivanje zahtjeva na strani korisnika*.



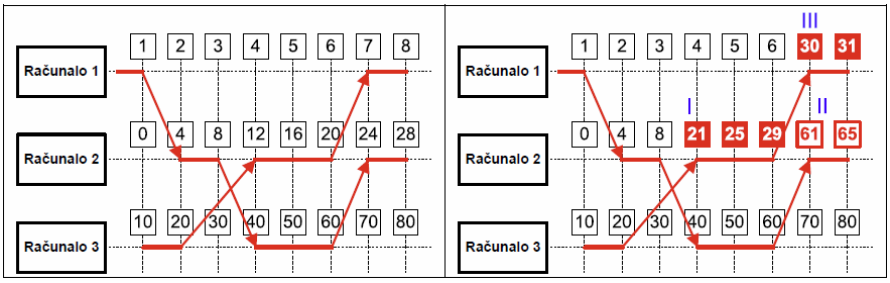
Objasnite načine na koje se indeks dokumenata može podijeliti između čvorova u raspodijeljenim tražilicama.

Indeks dokumenata se dijeli prema dokumentima ili riječima. U prvom slučaju je svaki čvor zadužen za podskup dokumenata iz kolekcije za koji izgrađuje invertirani indeks, a pri rješavanju upita kontaktira se svaki čvor koji generira odgovor na temelju vlastite kolekcije. U drugom slučaju je svaki čvor zadužen za dio rječnika kompletne kolekcije. Procesira je upita je jednostavnije nego u prethodnom slučaju, no problem je kreiranje i održavanje ovakvog raspodijeljenog indeksa.

Računski i praktični zadaci

Prvi ciklus

Za slijed razmijenjenih poruka između tri računala prikazan slikom, uspostavite globalni tijek vremena primjenom skalarnih oznaka logičkog vremena. Navedite i objasnite trenutke u kojima se ostvaruje korekcija lokalnih satnih mehanizama.

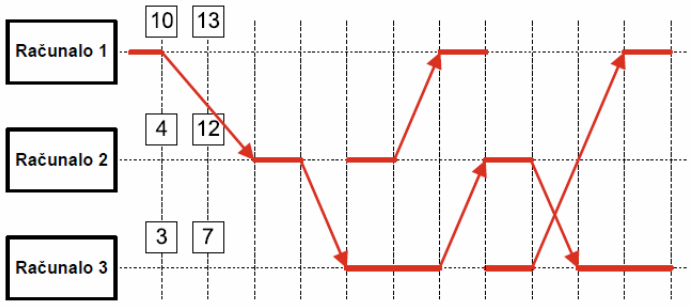


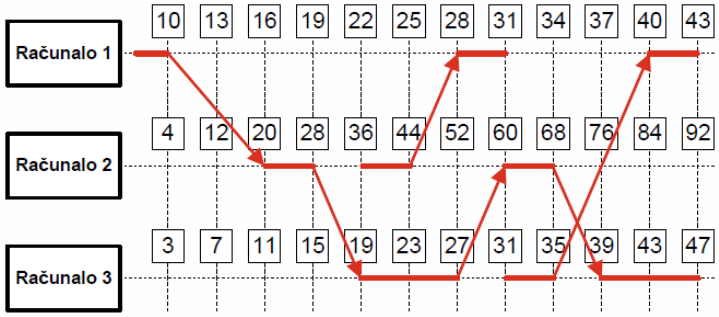
Trenutak I: Računalo 2 prima poruku od računala 3 s oznakom vremena T_P=20 koja je veća od lokalne oznake vremena T_L= 12. Lokalni sat se pomiče na vrijednost T_P+1=21.

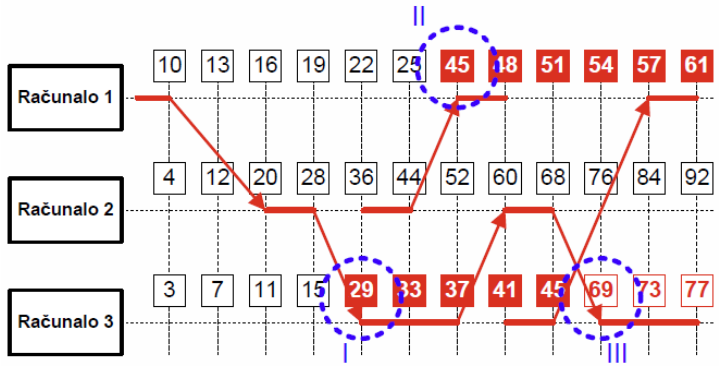
Trenutak II: Računalo 2 prima poruku od računala 3 s oznakom vremena T_P=60 koja je veća od lokalne oznake vremena T_L= 33. Lokalni sat se pomiče na vrijednost T_P+1=61.

Trenutak III: Računalo 1 prima poruku od računala 2 s oznakom vremena T_P=29 koja je veća od lokalne oznake vremena T_L= 7. Lokalni sat se pomiče na vrijednost T_P+1=30.

Za slijed razmijenjenih poruka između tri računala prikazan slikom, uspostavite globalni tijek vremena primjenom skalarnih oznaka logičkog vremena. Navedite i objasnite trenutke u kojima se ostvaruje korekcija lokalnih satnih mehanizama.





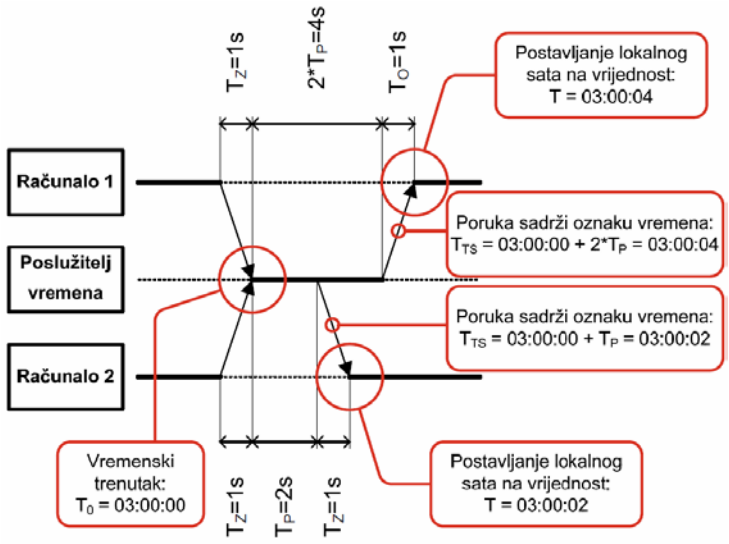


Trenutak I: Računalo 3 prima poruku od računala 2 s oznakom vremena T_P=28 koja je veća od lokalne oznake vremena T_L= 19. Lokalni sat se pomiče na vrijednost T_P+1=29.

Trenutak II: Računalo 1 prima poruku od računala 2 s oznakom vremena T_P=44 koja je veća od lokalne oznake vremena T_L= 28. Lokalni sat se pomiče na vrijednost T_P+1=45.

Trenutak III: Računalo 3 prima poruku od računala 2 s oznakom vremena T_P=68 koja je veća od lokalne oznake vremena T_L=49. Lokalni sat se pomiče na vrijednost T_P+1=69.

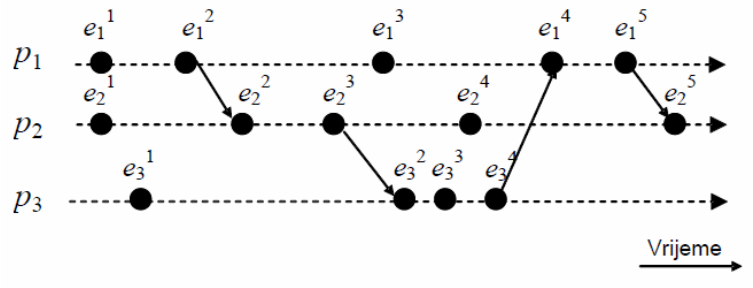
U raspodijeljenoj okolini nalazi se poslužitelj vremena i dva računala koja koriste poslužitelj primjenom algoritma Christian kako bi međusobno uskladili vrijednosti satnih mehanizama. Poruke zahtjeva i odgovora od svakog računala do poslužitelja putuju $T_Z=T_O=1s$ dok je vrijeme obrade poruke $T_P=2s$. U slučaju da poslužitelj istodobno primi dva ili više zahtjeva, poslužitelj slijedno obrađuje primljene zahtjeve. U trenutku 3:00:00 poslužitelj prima poruku zahtjeva od računala 2 te nakon 0 sekundi prima poruku zahtjeva od računala 1. Ukoliko pretpostavimo da svi satovi imaju isti takt, kolika je razlika vremena između sva tri računala u raspodijeljenoj okolini nakon završetka izvođenja algoritma za oba računala? Dopunite vremenski dijagram karakterističnim točkama.



Nakon što računalo 2 primi poruku dogovora s oznakom vremena 03:00:02, postavlja lokalni satni mehanizam na vrijednost 03:00:02. Nakon što računalo 1 primi poruku dogovora s oznakom vremena 03:00:04, postavlja lokalni satni mehanizam na vrijednost 03:00:04. Nakon što računalo 1 primi poruku odgovora i ostvari usklađivanje lokalnog sata, satni mehanizmi računala imaju sljedeće vrijednosti i odstupanja u odnosu na ostala računala:

R_1:	03:00:04,	ΔR_{12}:	00:00:00,	ΔR_{1P}:	+ 00:00:01
R_P:	03:00:05,	ΔR_{P1}:	– 00:00:01,	ΔR_{P2}:	– 00:00:01
R_2:	03:00:04,	ΔR_{21}:	00:00:00,	ΔR_{2P}:	+ 00:00:01

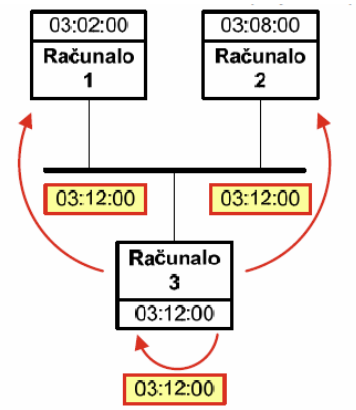
Na temelju primjera procesa sa slike objasnite jesu li sljedeći parovi događaja uzročno povezani ili nisu? a) e_1^3 i e_2^2 b) e_2^2 i e_1^5



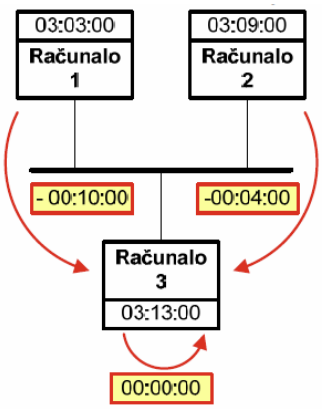
a) Događaji e_1^3 i e_2^2 su neovisni zato što nisu slijedni događaji na istom procesu, između njih ne postoji slanje i primanje poruke te između njih ne postoji tranzitivna uzročnost.

b) Između događaja e_2^2 i e_1^5 postoji uzročna povezanost preko tranzitivne uzročnosti e_2^2 -> $e_2^3 \wedge e_2^3$ -> $e_3^2 \wedge e_3^2$ -> $e_3^3 \wedge e_3^3$ -> $e_3^4 \wedge e_3^4$ -> $e_1^4 \wedge e_1^4$ -> e_1^5.

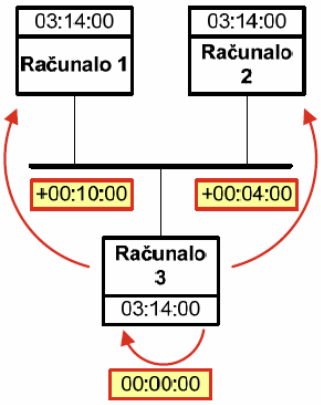
Prikažite i objasnite korake algoritma Berkeley za usklađivanje satnih mehanizama tri računala u raspodijeljenoj okolini. Računala imaju sljedeće vrijednosti satova T1=03:02:00, T2=03:08:00 i T3=03:12:00. Upravitelj je treće računalo. Pretpostavite da prijenos poruke između 2 računala traje 1 minutu.



Upravitelj šalje poruku s trenutnim vremenom svim računalima.

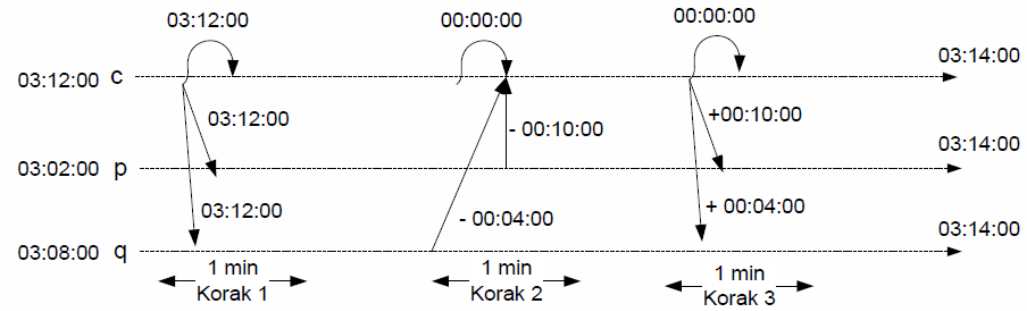


Poslane poruke putuju 1 minutu i nakon primitka poruka, računala odgovaraju s porukom koja sadrži razliku lokalnog vremena u odnosu na primljeno vrijeme.



Nakon primitka poruka odgovora, upravitelj šalje poruke zahtjeva koje sadrže vremenski pomak za svako računalo. Poruke zahtjeva putuju 1 minutu te nakon primitka poruke zahtjeva, svako računalo usklađuje lokalni satni mehanizam.

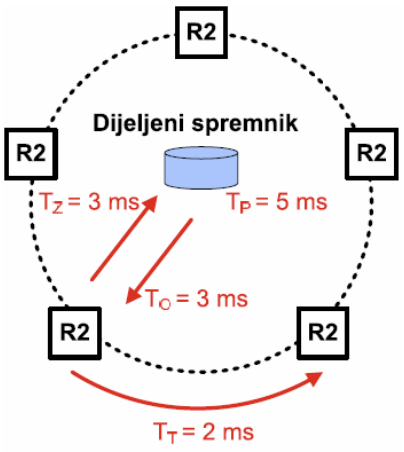
Noviji dijagram za isti zadatak:



Pet procesa postavljenih na različita računala u raspodijeljenoj okolini ostvaruje međusobno isključivanje primjenom prstena. Vrijeme prijenosa poruke zahtjeva i odgovora pri pristupu dijeljenom sredstvu jednako je 3 ms, vrijeme obrade poruke zahtjeva na sredstvu je 5 ms, vrijeme prijenosa tokena između dva susjedna procesa u prstenu je 2 ms. Kada primi token, proces može maksimalno jednom ostvariti pristup dijeljenom sredstvu prije nego što proslijedi token idućem susjedu. Prikažite strukturu prstena i naznačite navedena vremena na slici. Koje je minimalno, a koje maksimalno vrijeme čekanja bilo kojeg procesa u prstenu za pristup dijeljenom sredstvu.

Minimalno vrijeme čekanja na pristup

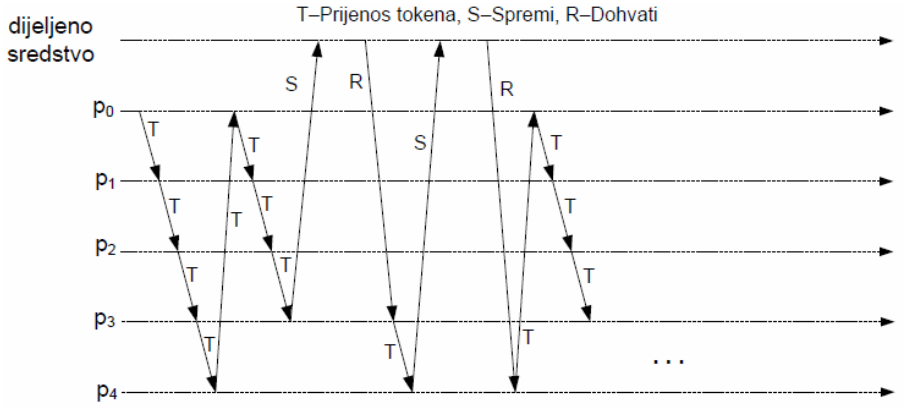
U najboljem slučaju, proces koji želi ostvariti pristup čeka T=0 sekundi. Naime, taj slučaj nastupa kada proces uđe u stanje u kojem želi ostvariti pristup sredstvu netom prije nego što je primio token.



Maksimalno vrijeme čekanja na pristup

U najgorem slučaju, proces ulazi u stanje u kojem želi ostvariti pristup sredstvu netom nakon što je proslijedio token svojem susjedu. U tom slučaju, proces mora čekati da svi ostali procesi prime token i ostvare pristup dijeljenom sredstvu. Maksimalno vrijeme čekanja u tom slučaju iznosi
$T = 5 * T_T + 4 * (T_Z + T_O + T_P) = 10 + 44 = 54$ ms.

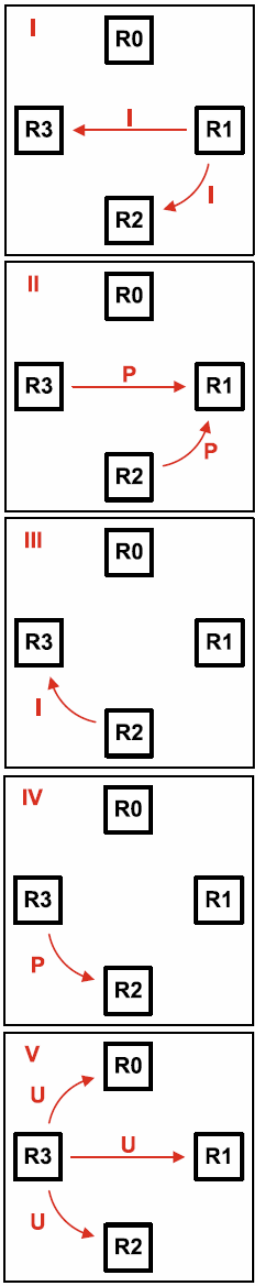
Noviji dijagram za isti zadatak:



Min. vrijeme - U najboljem slučaju, proces koji želi ostvariti pristup čeka T=0 sekundi. Naime, taj slučaj nastupa kada proces uđe u stanje u kojem želi ostvariti pristup sredstvu netom prije nego što je primio token.

Max. vrijeme - U najgorem slučaju, proces ulazi u stanje u kojem želi ostvariti pristup sredstvu netom nakon što je proslijedio token svojem susjedu. U tom slučaju, proces mora čekati da svi ostali procesi prime token i ostvare pristup dijeljenom sredstvu. Maksimalno vrijeme čekanja u tom slučaju iznosi
$T = 5 * T_T + 4 * (T_Z + T_O + T_P) = 10 + 44 = 54$ ms.

Prikažite postupak određivanja upravitelja između četiri računala (R_0, R_1, R_2, R_3) u raspodijeljenoj okolini. Postupak odlučivanja započinje računalo R_1.



Postupak izbora započinje računalo R_1 slanjem poruke *izbor* (I) svim računalima većeg rednog broja od računala R_1. Poruka izbora poslana je računalima R_2 i R_3.

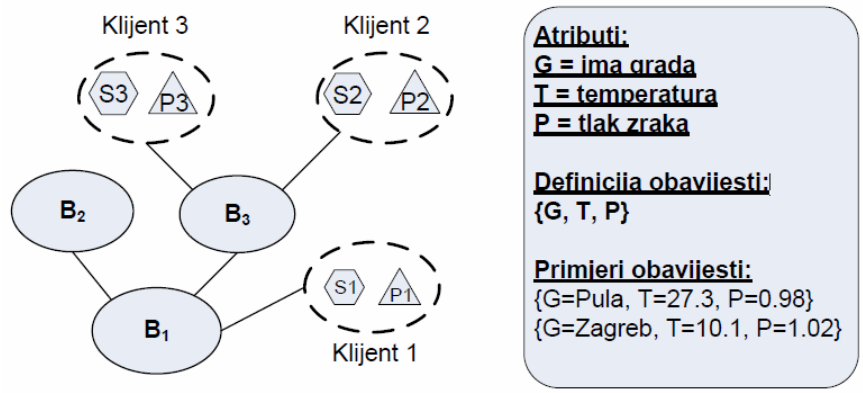
Računala R_2 i R_3 primaju poruku izbora. Obzirom da su oba računala aktivna, oba računala šalju poruku *potvrda* (P) računalu R_1.

Obzirom da je računalo R_3 računalo s najvećim rednim brojem, računalo R_3 niti jednom računalu ne šalje poruku *izbor*. Međutim, računalo R_1 nije računalo s najvećim rednim brojem te stoga šalje poruku *izbor* (I) računalu R_3.

Računalo R_3 prima poruku *izbor* (I) od računala R_2 i šalje poruku *potvrda* (P) računalu R_2.

Obzirom da je računalo R_3 računalo s najvećim rednim brojem, računalo R_3 postaje novi upravitelj te šalje poruku *upravitelj* (U) svim računalima (R_0, R_1, R_2).

Raspodijeljeni sustav objavi-pretplati, u kojem se koristi algoritam preplavljivanja obavijestima, sastoji se od 3 posrednika i 3 klijenta kako je prikazano slikom. Svaki klijent u sustavu ima ulogu pretplatnika i objavljivača. Odgovorite na sljedeća pitanja:

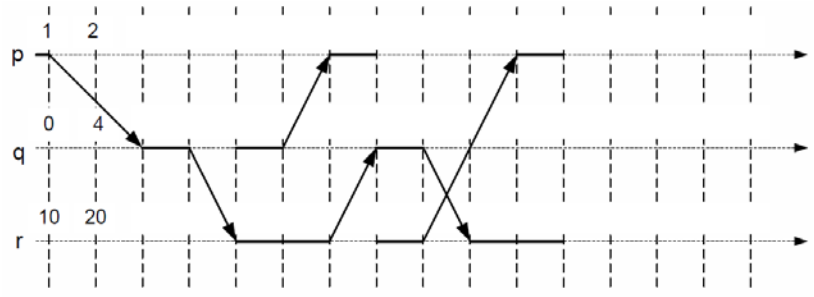


a) U trenutku t1 **klijent 1** generira pretplatu ***s1={G=Zagreb,T<15.5,P>0.98}***. Napišite oznake svih posrednika na kojima se pohranjuje ova pretplata. **Pretplata se pohranjuje na posredniku B1.**

b) U trenutku t2>t1 **klijent 2** generira pretplatu ***s2=s1***. Napišite oznake svih posrednika na kojima se pohranjuje ova pretplata. **Pretplata se pohranjuje na posredniku B3.**

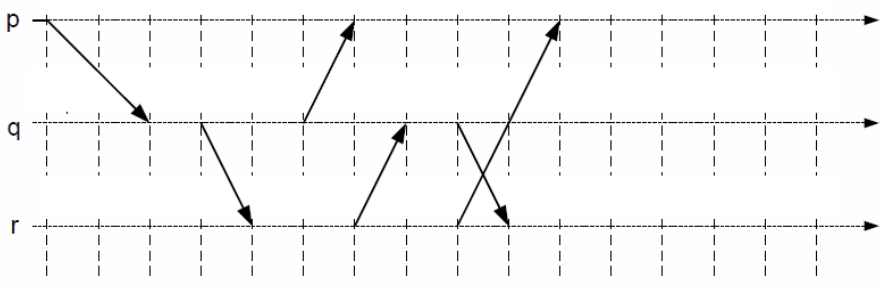
c) U trenutku t3>t2 **klijent 3** generira obavijest ***p1={G=Zagreb,T=-2.2,P=1.01}***. Objasnite točan redoslijed kojim de se ova obavijest proširiti sustavom i biti isporučena zainteresiranim klijentima.
P3->B3(->S2)->B1(->S1)->B2

Za slijed razmijenjenih poruka između tri računala prikazan slikom, uspostavite globalni tijek vremena primjenom skalarnih oznaka logičkog vremena. Navedite i objasnite trenutke u kojima se ostvaruje korekcija lokalnih satnih mehanizama.



***

Ukratko objasnite razliku između skalarnih i vektorskih oznaka vremena prilikom sinkronizacije procesa u vremenu te navedite prednosti i nedostatke jedne i druge metode. Za slijed razmijenjenih poruka između tri računala prikazan slikom, uspostavite tijek akcija primjenom vektorskih oznaka logičkog vremena.



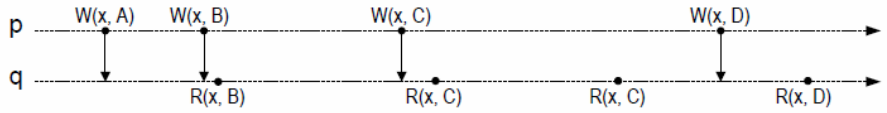
***

Drugi ciklus

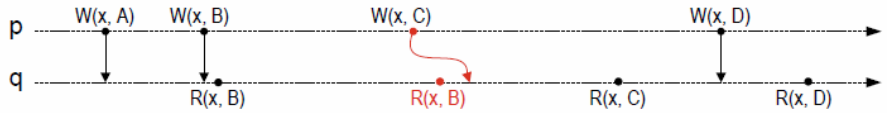
Objasnite što je <u>stroga konzistentnost</u> operacija u raspodijeljenim sustavima? Na primjeru procesa p i q prikažite slijed operacija čitanja i pisanja koji je a) u skladu i b) nije u skladu s načelima stroge konzistentnosti.

Model nalaže da su sve operacije pisanja trenutno vidljive svim ostalim procesima u raspodijeljenoj okolini. Naime, bez obzira koliko je kratko vremena prošlo od posljednje operacija pisanja, ako se provede operacija čitanja na bilo kojem računalu rezultat de biti upravo sadržaj zapisan u prethodnoj operaciji pisanja.

a)



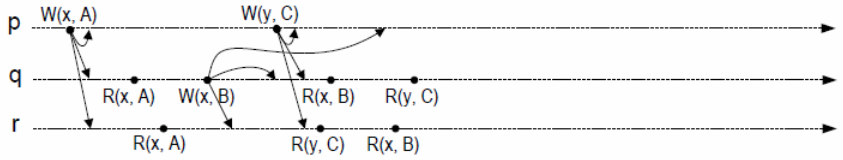
b)



Objasnite što je <u>povezana konzistentnost</u> operacija u raspodijeljenim sustavima? Na primjeru procesa p, q i r prikažite slijed operacija čitanja i pisanja koji je a) u skladu i b) nije u skladu s načelima povezane konzistentnosti.

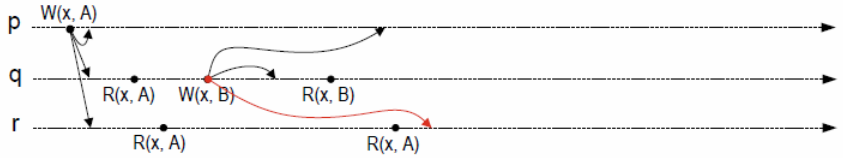
Redoslijed izvođenja povezanih operacija pisanja vidljiv je svim procesima na jednak način, dok redoslijed izvođenja operacija pisanja koje nisu povezane svakom procesu može biti prikazan na drugačiji način.

a)



Čitanje q: A->B->C
Čitanje r: A->C->B

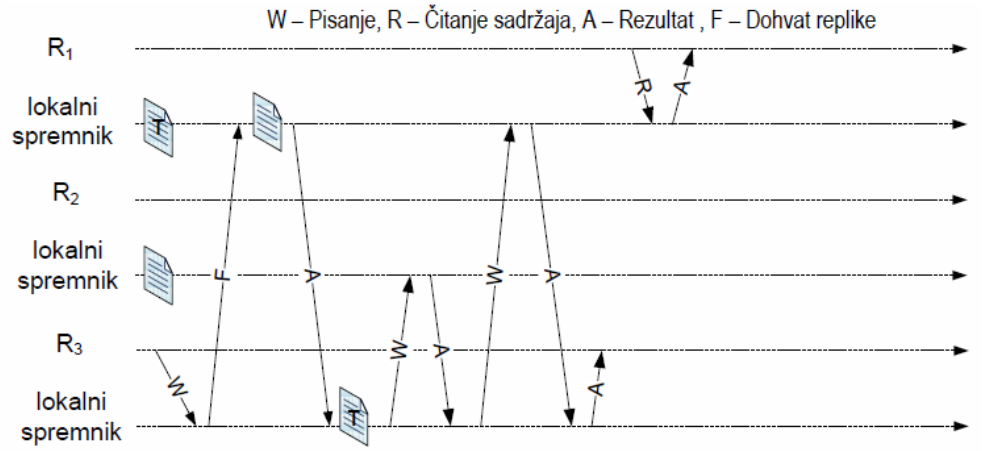
b)



Čitanje q: A->B
Čitanje r: A->A

Raspodijeljeni sustav uključuje tri računala (*R_0*, *R_1*, *R_2*) s lokalnim spremnicima. U lokalnom spremniku računala *R_1* nalazi se trajna replika dokumenta, dok se u lokalnom spremniku računala *R_2* nalazi obična replika dokumenta. Korisnik putem računala *R_3* provodi operaciju pisanja nad dokumentom primjenom postupka lokalnog obnavljanja stanja replike. Skicirajte i objasnite korake postupka.

R_3 upućuje zahtjev za provođenje operacije pisanja lokalnom spremniku sustava replika. Obzirom da to računalo u lokalnom spremniku ne sadrži trajnu repliku, računalo dohvaća trajnu repliku s R_1 u svoj lokalni spremnik. Replika na R_1 se pretvara u običnu repliku. Nakon prenošenja trajne replike u lokalni spremnik od R_3, obnavlja se njeno stanje i prosljeđuje se ostalim računalima u sustavu. Nakon što ta računala pošalju potvrdu, R_3 prosljeđuje potvrdu korisniku koji je započeo operaciju pisanja. Ostali korisnici zatim imaju mogućnost provođenja operacije čitanja nad najbližom replikom u sustavu replika.



U sustavu replika koji se sastoji od glavnog poslužitelja i n=4 podjednako opterećena pomoćna poslužitelja, odredite metodu održavanja konzistentnosti replika za koju će prosječno mrežno (prometno) opterećenje poslužitelja L biti najmanje. Pri tome pretpostavite da korisnike isključivo poslužuju pomoćni poslužitelji, da je prosječna frekvencija upita f_u=5 upita/s, prosječna frekvencija promjena f_p=1 promjena/min te da su prosječne veličine upita/odgovora, operacija za promjenu sadržaja i replika l_p=1kB, l_o=50 kB i l_r=100 kB. Usporedite dobivena opterećenja s centraliziranim slučajem kada korisnike poslužuje glavni poslužitelj.

PUSH metoda s prosljeđivanjem obavijesti o promjenama: Nakon svake promjene, glavni poslužitelj šalje svakom pomoćnom poslužitelju obavijest o promjenama. Za prvi sljedeći korisnički zahtjev, svaki od pomoćnih poslužitelja šalje glavnom poslužitelju upit za novom replikom na koji mu glavni poslužitelj odgovara novom verzijom replike.
$\mathbf{L_1 = n \cdot f_p \cdot l_p + n \cdot f_p \cdot l_p + n \cdot f_p \cdot l_r} = n \cdot f_p \cdot (2 \cdot l_p + l_r) = 6{,}8$ kB/s

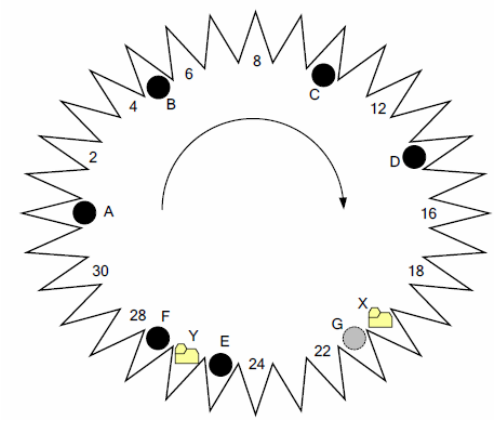
PUSH metoda s prosljeđivanjem operacija za promjenu sadržaja: Nakon svake promjene, glavni poslužitelj šalje svakom pomoćnom poslužitelju obavijest o operacijama koje je potrebno izvršiti nad prethodnom verzijom replike da bi se ona dovela u konzistentno stanje.
$\mathbf{L_2 = n \cdot f_p \cdot l_o} = 3{,}33$ kB/s

PUSH metoda s prosljeđivanjem cjelokupnog sadržaja replika: Nakon svake promjene, glavni poslužitelj šalje novu verziju replike svakom pomoćnom poslužitelju.
$\mathbf{L_3 = n \cdot f_p \cdot l_r} = 6{,}67$ kB/s

PULL metoda: Nakon svakog upita, pomoćni poslužitelj provjerava kod glavnog poslužitelja je li došlo do promjene stanja replike koju pohranjuje lokalno. U 1/(n·fp) od 1/fu slučajeva će poslužitelj odgovoriti novom verzijom replike, a u 1/(fu - n·fp) od 1/fu slučajeva će odgovoriti porukom da nije došlo do promjene.
$\mathbf{L_4 = f_u \cdot l_p + n \cdot f_p \cdot l_r + (f_u - n \cdot f_p) \cdot l_p} = 16{,}6$ kB/s

Centralizirani slučaj: Glavni poslužitelj odgovara replikom na svaki korisnički upit.
$\mathbf{L_5 = f_u \cdot (l_p + l_r)} = 505$ kB/s

Na slici je prikazana mreža Chord koja se sastoji od 6 čvorova (A, B, C, D, E i F) i koristi prostor identifikatora duljine N=32 (dovoljno je m=5 bita za kodiranje). Ukoliko je H1(A)=0, H1(B)=5, H1(C)=10, H1(D)=14, H1(E)=25 i H1(F)=27, odgovorite na sljedeća pitanja:



1. Popunite tablice usmjeravanja čvorova A i F.

Routing table *A*(0)
1, 5
2, 5
4, 5
8, 10
16, 25

Routing table *F*(27)
28, 0
29, 0
31, 0
3, 5
11, 14

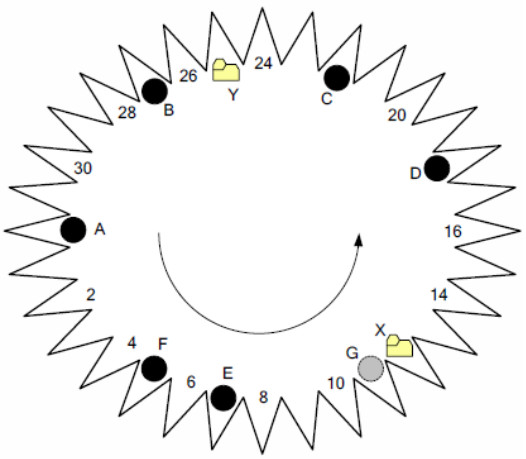
2. Na kojem će se čvoru pohraniti podatak X s ključem H2(X)=20?
Na čvoru E(25).

3. Odredite slijed čvorova preko kojih se usmjerava upit od čvora A s ciljem pronalaska podatka Y s ključem H2(Y)=26.
A-E-F

4. Dodan je novi čvor G (H1(G)=21) u mrežu.
Što će se promijeniti u tablici usmjeravanja čvora A?
Umjesto 16, 25 pisat će 16, 21.

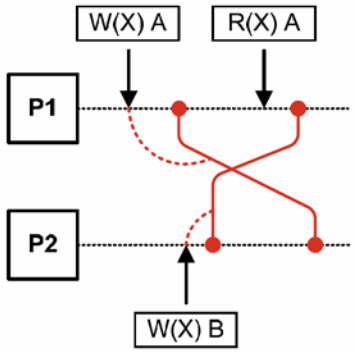
Na slici je prikazana mreža <u>Chord</u> koja se sastoji od 6 čvorova (*A, B, C, D, E* i *F*) i koristi prostor identifikatora veličine *N=32* (dovoljno je *m=5* bita za kodiranje). Ako su čvorovima pridijeljeni sljedeći ključevi je $H_1(A)=0$, $H_1(B)=27$, $H_1(C)=22$, $H_1(D)=18$, $H_1(E)=7$ i $H_1(F)=5$, odgovorite na sljedeća pitanja:

1. Popunite tablice usmjeravanja čvorova *A* i *F*.
2. Na kojem de se čvoru pohraniti podatak *X* s ključem $H_2(X)=12$?
3. Odredite slijed čvorova preko kojih se usmjerava upit od čvora *A* s ciljem pronalaska podatka *Y* s ključem $H_2(Y)=25$.
4. Dodan je novi čvor *G* ($H_1(G)=11$) u mrežu. Što de se promijeniti u tablici usmjeravanja čvora *A*?
5. Popunite tablicu usmjeravanja čvora G.

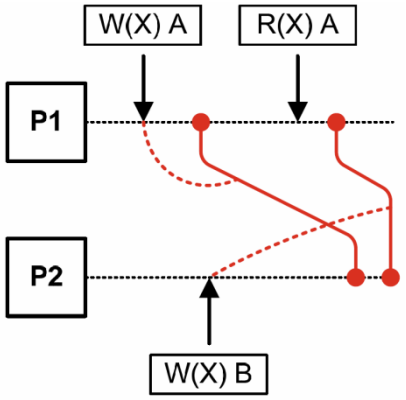


***

Na slici je prikazan redoslijed izvođenja operacija dvaju procesa. Objasnite poštuje li prikazani slijed izvođenja operacija slijednu konzistentnost? Ako da, prikazani primjer promijenite tako da narušite slijednu nekonzistentnost izvođenja operacija. U suprotnom prikazani primjer promijenite tako da ostvarite slijednu nekonzistentnost izvođenja operacija. Obrazložite predloženo rješenje.

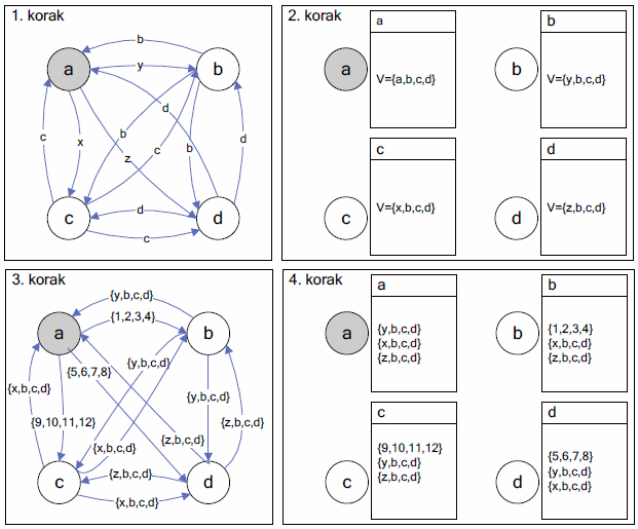


Slijedna konzistentnost nalaže da je redoslijed izvođenja operacija nebitan. Međutim, svi procesi moraju na jednak način vidjeti redoslijed izvođenja operacija u vremenu. U prikazanom primjeru nije ostvarena slijedna konzistentnost izvođenja operacija. Naime, proces P1 vidi da se na lokaciju X prvo zapisuje vrijednost A, te zatim vrijednost B, dok proces P2 vidi da se na lokaciju X prvo zapisuje vrijednost B a zatim vrijednost A.

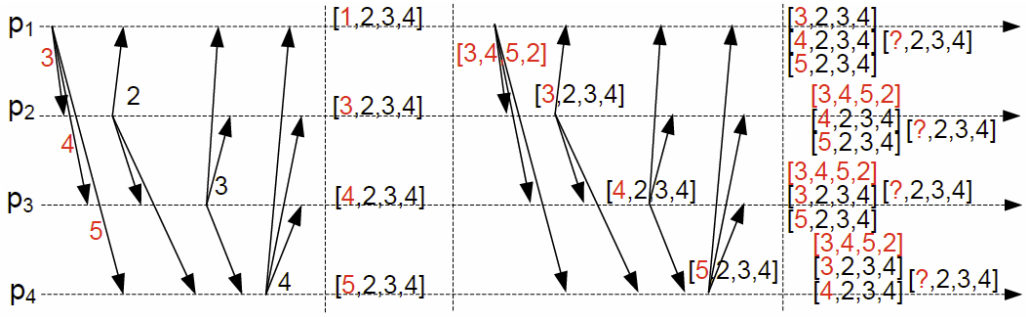


Da bi se ostvarila slijedna konzistentnost, potrebno je osigurati da svi procesi vide na jednak način redoslijed izmjena sadržaja lokacije X u vremenu. Primjerice, odgađanjem učinaka provođenja operacije pisanja na lokaciju X sadržaja B koju provodi proces P2, na način prikazan na slici .

U grupi od 4 procesa s identifikatorima a, b, c i d proces a je neispravan (pretpostavite bizantski ispad). Grupa procesa želi postidi sporazum o identifikatorima ostalih procesa grupe. U koracima 1 i 3 procesi međusobno razmjenjuju podatke, a u koracima 2 i 4 prikupljaju i analiziraju primljene podatke. Nacrtajte na slici koje podatke procesi razmjenjuju u koracima 1 i 3 (nacrtajte strelice i razmijenjene podatke uz svaku strelicu), a za korake 2 i 4 navedite podatke (napišite ih u pripremljene kućice) koje pojedini proces ima na raspolaganju radi donošenja odluke o sporazumu.



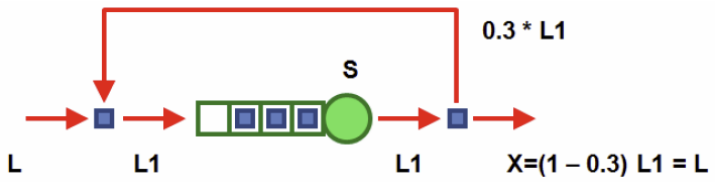
U grupi od 4 procesa (p_1, p_2, p_3 i p_4) proces p_1 je neispravan (pretpostavite bizantski ispad). Grupa procesa želi postići sporazum o identifikatorima ostalih procesa grupe. U koracima 1 i 3 procesi međusobno razmjenjuju podatke, a u koracima 2 i 4 prikupljaju i analiziraju primljene podatke. Nacrtajte na slici podatke koje procesi razmjenjuju u koracima 1 i 3, a za korake 2 i 4 navedite podatke koje pojedini proces ima na raspolaganju radi donošenja odluke o sporazumu.



Sustav sadrži 3 serijske procesne jedinice s prosječnim vremenima posluživanja 1 s, 2 s i 3 s. Koliko de biti vrijeme zadržavanja u sustavu uz ulazni ritam zahtjeva od 0,1 z/s? Koliki će biti prosječni broj zahtjeva u sustavu?

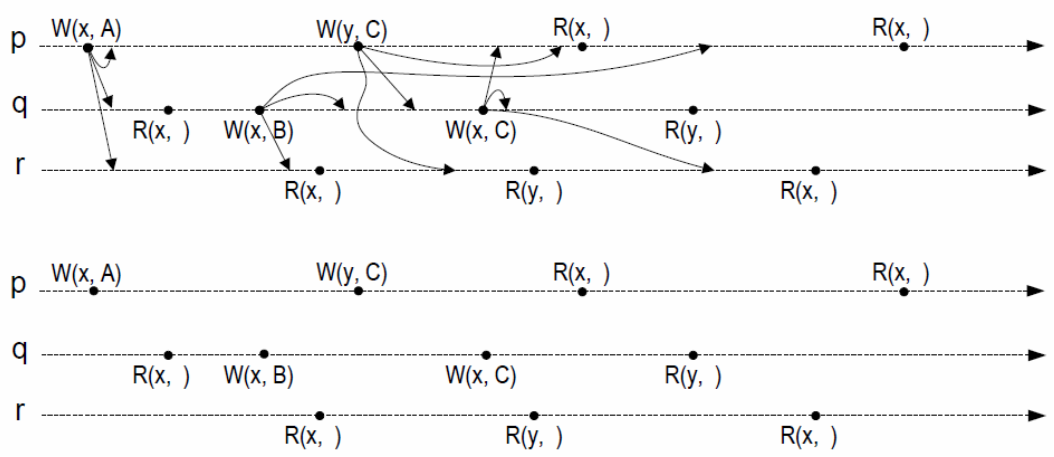
Prosječna vremena posluživanja su S1 = 1 s/z, S2 = 2 s/z, S3 = 3 s/z.
U stabilnom stanju sustava je X = L.
Propusnost sustava je X = 0.1 z/s.
Vremena zadržavanja je $R_N = S_N / (1 – X*S_N)$.
$R_1 = 1.11s$, $R_2 = 2.5s$, $R_3 = 4.29s$
Prosječni broj zahtjeva u repu je $Q = X * (R_1 + R_2 + R_3) = 0.1$ z/s * (1.11 + 2.5 + 4.29) = 0.79 z.

Paketi dolaze u komunikacijski kanal s učestalošću 0,5 paketa u sekundi i zahtijevaju 0,75 sekundi za obradu. Za 30 % paketa dogodi se pogreška pri prijenosu i takvi paketi se umeću u rep za ponovno slanje. Koliko vremena paket prosječno provede u kanalu?



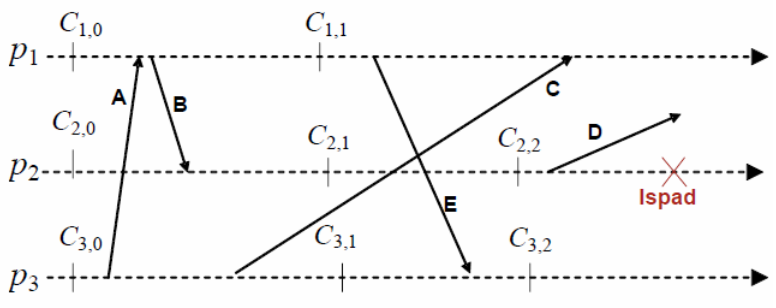
- Broj pristiglih paketa u sekundi je L = 0.5 p/s.
- Prosječno vrijeme obrade paketa je S = 0.75 s/p.
- Vjerojatnost pogreške paketa pri prijenosu je p = 0.3.
- L1 = p L1 + L => L1 = L / (1 – p)
- L1 = L / (1 – p) = 0.5 / 0.7 = 0.714 p/s
- Prosječna zaposlenost kanala je U = L1 * S = 0.714 p/s * 0.75 s/p = 0.536 (53.6 %).
- Srednje vrijeme čekanja u repu je W = S * U / (1 – U) = 0.866 s/p.
- Srednje vrijeme zadržavanja paketa u kanalu je R1 = W + S = 0.866 s/p + 0.75 s/p = 1.616 s/p.
- Prosječno vrijeme koje paket provede u kanalu je R = R1 / (1 – p) = 2.31.

Na slici je prikazan redoslijed izvođenja operacija triju procesa. Objasnite poštuje li prikazani slijed izvođenja operacija konzistentnost redoslijeda? Ako da, prikazani primjer promijenite tako da narušite konzistentnost redoslijeda izvođenja operacija. U suprotnom prikazani primjer promijenite tako da ostvarite konzistentnost redoslijeda izvođenja operacija. Obrazložite predloženo rješenje.



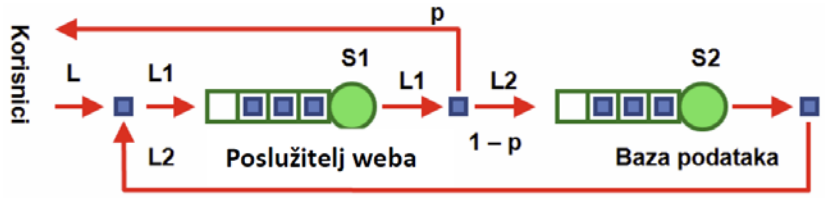
***

Sustav na slici koristi strategiju oporavka unazad, a svaki proces bilježi kontrolne točke neovisno. Navedite konzistentno stanje u koje je sustav potrebno vratiti nakon ispada. Što će se dogoditi s porukama C i D nakon što se sustav vrati u konzistentno stanje i započne s daljnjim izvođenjem.



***

Web poslužitelj opisan je modelom zasnovanim na repovima koji je prikazan na slici. Vjerojatnost usmjeravanja zahtjeva prema korisniku (p) iznosi 0.9. Ako obrada zahtjeva na poslužitelju weba u prosjeku traje 0.01 sekundu, dok obrada zahtjeva na bazi podataka u prosjeku traje 0.1 sekundu. Koliko je vrijeme zadržavanja u sustavu uz ulazni ritam dolazaka zahtjeva u iznosu od 1 zahtjev u sekundi.

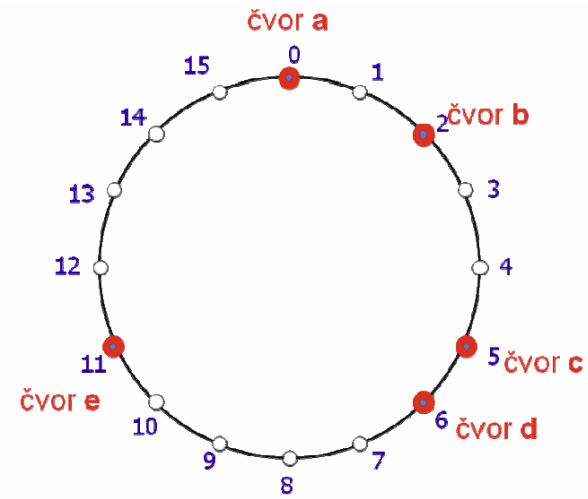


***

Opišite značajke raspoređivanja zasnovanog na korištenju <u>prostorne lokalnosti</u> i <u>vremenske lokalnosti</u>. Skicirajte primjer obje vrste raspoređivanja.

***

U strukturiranom P2P sustavu hash tablica ključeva koji omogućuju pronalaženje podataka raspodijeljena je na 5 čvorova (a, b, c, d, e) u prostoru identifikatora veličine N = 16, kojima su dodijeljeni identifikatori prema funkciji H1 kako slijedi:
Čvor a: 0, Čvor b: 2, Čvor c: 5, Čvor d: 6 i Čvor e: 11.



Ukoliko se neki podatak pridijeli funkcijom H2 identifikatoru 8, objasnite na koji će se čvor taj podatak pohraniti?
Kako ne postoji čvor s identifikatorom 8, podatak će se pohraniti na prvi sljedeći čvor, a to je čvor e s identifikatorom 11.

Disk za trajno spremanje podataka ispunjava 50 zahtjeva u sekundi. Srednje vrijeme obrade zahtjeva operacija pisanja i čitanja je 10 ms. Disk ima prosječno 1 zahtjev u repu. Koliko je prosječno vrijeme čekanja na obradu zahtjeva?

Srednje vrijeme obrade zahtjeva je S = 10 ms/z.
Propusnost sustava je X = 50 z/s.
Broj zahtjeva u repu je Q = 1 z.
Vrijeme zadržavanja zahtjeva u sustavu je R = Q / X = (1 z) / (50 z/s) = 20 ms.
Vrijeme zadržavanja R uključuje vrijeme čekanja u repu (W) i vrijeme obrade zahtjeva (S): R = W+S.
Vrijeme čekanja na obradu je W = R – S = 20 ms – 10 ms = 10 ms.

Web aplikacija uključuje podršku korisnicima putem chat usluge. Kupci sami odabiru jedan od 10 repova čekanja. Mjerenja pokazuju da zahtjevi prosječno dolaze 3 upita u minuti te da svaki kupac prosječno čeka 3 minute u repu i prosječno provodi 2 minute u konverzaciji. Koliko je srednje vrijeme zadržavanja kupaca za zadani sustav?

Prosječno vrijeme posluživanja je S = 2 min/z.
Broj pristiglih zahtjeva u jednom repu je L = 3 z/min.
U stabilnom stanju sustava X = L.
Prosječna zaposlenost sustava je U = S L = (2 min/z) (3 z/min) = 6.
Faktor iskorištenja je ro = U / N = 6 / 10 = 0,6.
Srednje vrijeme zadržavanja korisnika u sustavu je R = S / (1 - ro) = 2 / (1 – 0.6) = 5 min.
Srednje vrijeme čekanja u repu je W = R – S = 5 min – 2 min = 3 min.

Pretpostavite da grupa procesa treba postidi sporazum. U slučaju da su dva procesa grupe u stanju bizantskog ispada, koji je minimalni ukupni broj procesa u grupi za postizanje sporazuma?

k = 2, N = 3k+1 = 7.