[naziv tvrtke]

[naslov dokumenta]

[podnaslov dokumenta]

Sadržaj

7. Otpornost na neispravnosti u raspodijeljenom okružju 2

8. Računarstvo u oblaku i performanse 4

9. Mikrousluge (mikroservisi), komunikacija temeljena na događajima 11

10. Replikacija podataka 14

11. Particioniranje i konzistentnost podataka 16

12. Sustavi s ravnopravnim sudionicima 18

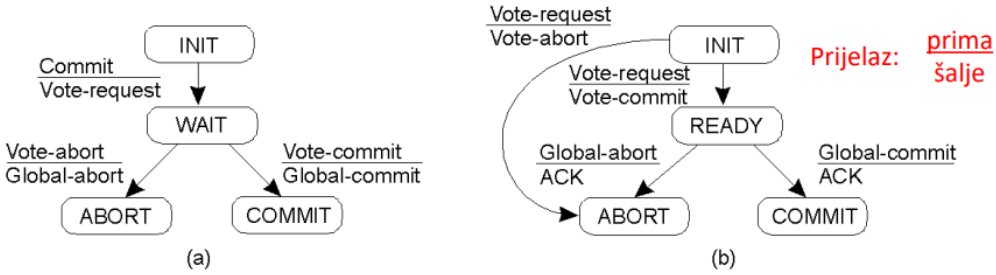
13. Tehnologija raspodijeljene glavne knjige 19

7. Otpornost na neispravnosti u raspodijeljenom okružju

Pretpostavite da postoji skupina od m jednakih procesa. Navedite toleranciju ove skupine procesa na ispade ako pretpostavite „obični" ispad ili bizantski ispad.

Tolerancija je osigurana s **k+1** procesa za ispad k procesa.

Identificirajte i objasnite blokirajuća stanja protokola 2PC.



Automat stanja koordinatora

Automat stanja procesa

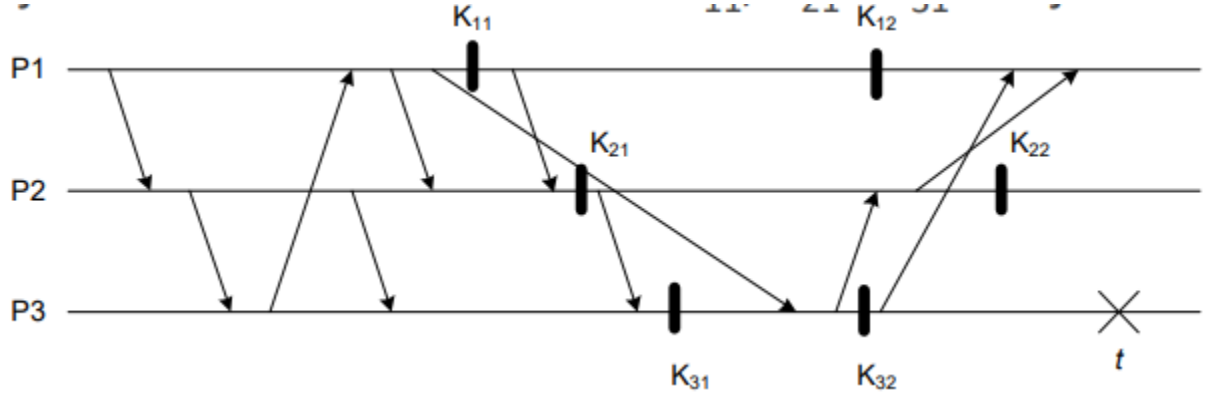
Proces je blokiran u stanju INIT kada čeka VOTE_REQUEST, ako ne primi poruku nakon određenog vremena, proces može lokalno odustati od izvršavanja operacije.

Proces je blokiran u stanju READY čekajući konačnu odluku koordinatora, proces treba saznati koju je poruku koordinator poslao i pitati druge procese što se događa.

Koordinator je blokiran u stanju WAIT kada čeka odgovore svih procesa, ako nakon nekog perioda ne primi odgovor od svih procesa, koordinator može zaključiti da treba odustati od izvršavanja operacije i poslati svim procesima GLOBAL_ABORT

2PC je blokirajući protokol, jer u slučaju ispada koordinatora nakon slanja VOTE_REQUEST, procesi ne mogu zaključiti o sljedećoj operaciji koju trebaju provesti (svi su u stanju READY)!

Slika prikazuje tri procesa i razmjenu poruka među njima. Svaki proces neovisno o drugi procesima bilježi svoja stanja u označenim kontrolnim točkama. U trenutku t dolazi do ispada procesa P3. Možemo li sustav od tri procesa na slici nakon ispada procesa P3 vratiti u konzistentno stanje koristeći kontrolne točke K11, K21 i K31 i objasnite zašto je to moguće ili nije moguće?



?

8. Računarstvo u oblaku i performanse

Koja je razlika između HPC-a (High-Performance Computing) i HTC-a (High-Thtoughput Computing)?

HTC radi kontinuiranu obradu velike količine raznovrsnih i nepovezanih podataka dok HPC radi paralelnu obradu velike količine sličnih i povezanih zadataka u kratkom periodu.

Što je grozd?

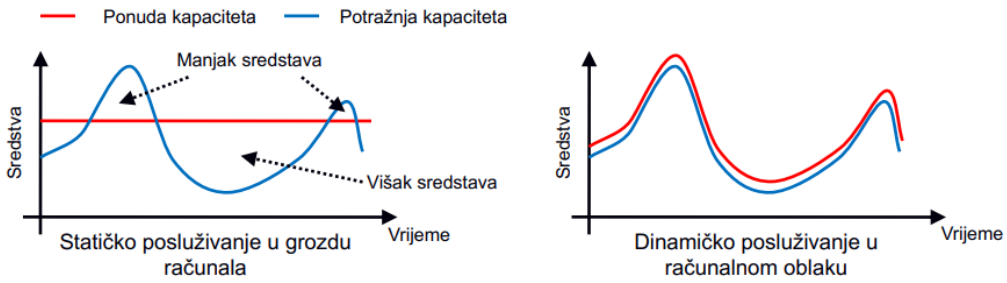
Skup samostalnih računala koje rade kooperativno i čine jedan jedinstveni računalni resurs. Računala su međusobno povezana lokalnom mrežom.

Što je splet?

Skup međusobno udaljenih heterogenih računalnih sredstava koje su međusobno (slabo) povezana Internetom. Rade na zajedničkom zadatku radi postizanja zajedničkog cilja.

Što je oblak?

Uslužno računarstvo, računalni resursi se iznajmljuju po potrebi i te resurse plaćamo onoliko koliko se koriste („pay as you go“). Prividno postoje neograničeni računalni resursi.



Usporedite grozd, splet i oblak po arhitekturi, resursima i aplikacijama.

	Grozd (*Cluster*)	Splet (*Grid*)	Oblak (*Cloud*)
Arhitektura	Skup računala povezanih brzom lokalnom mrežom	Skup udaljenih računala (ili grozdova) povezanih putem Interneta	Virtualizirani grozd računala koja se nalaze u jednom ili više podatkovnih centara
Resursi	Identična ili vrlo slična računala	Raznolika računala	Identična ili vrlo slična računala
Aplikacije	HPC, tražilice	Raspodijeljeno rješavanje problema	Uslužno računarstvo (*utility computing*), pohrana podataka
Primjeri	Google search engine, Cray XK7, BlueGene/Q, itd.	Folding@home, BOINC, SETI@home, itd.	Microsoft Azure, Amazon EC2, Google App Engine, itd.

Koja je razlika između iznajmljivanju resursa u oblaku i kada kupujemo vlastita računala?

U oblaku možemo jednostavno dodavati i smanjivati resurse ovisno o našoj potrebi, kod vlastitog računala u startu moramo odrediti koje ćemo komponente koristiti i nadogradnja zahtjeva kupnju i zamjenu komponenti. O vlastitom računalo brinemo sami (OS, storage, konfiguracija) dok u oblaku, ovisno o tipu usluge, o tome brine pružatelj usluge.

Koje su vrste oblaka po smještaju i svrsi?

Javni oblak (public) - iznajmljuje se za javnu uporabu i u vlasništvu je organizacije koja prodaje usluge u oblaku.

Privatni oblak (private) - u privatnom vlasništvu poduzeća i samo to poduzeće ga koristi, ne smatra pravim „oblakom".

Zajednički oblak (community) - nekoliko organizacija dijeli jednu infrastrukturu.

Hibridni oblak (hybrid) - kompozicija 2 ili više oblaka različitih vrsta.

Koja su dva osnovna koncepta u računarstvu u oblaku te ih objasnite?

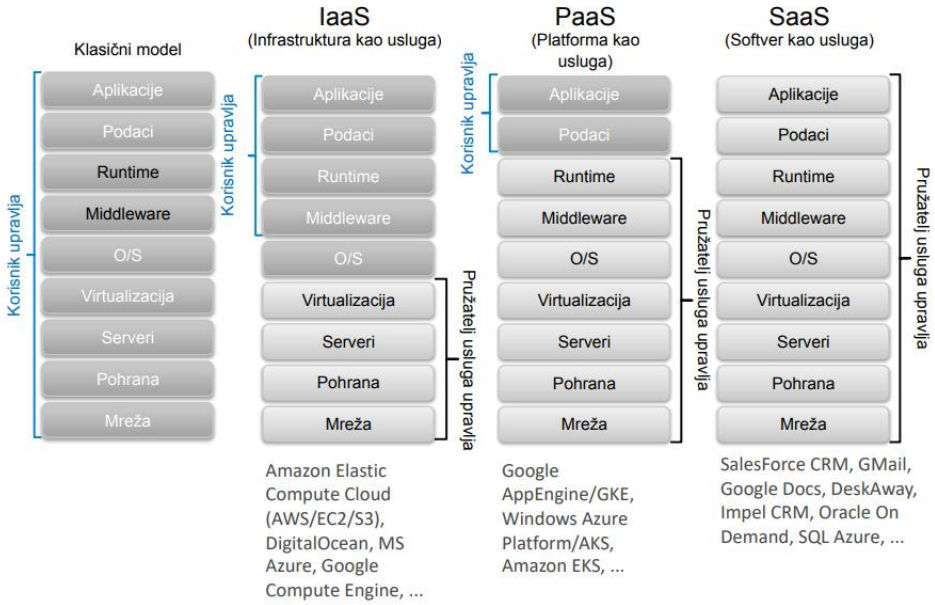
Apstrakcija implementacije

- Korisnik i razvijatelj ne znaju specifikaciju sustava na kojem će se aplikacije i usluge izvoditi
- Podaci spremljeni na lokacije koje nisu poznate
- Administracija sustava nije pod kontrolom razvijatelja
- Pristup aplikacijama i uslugama je omogućen putem Interneta

Virtualizacija

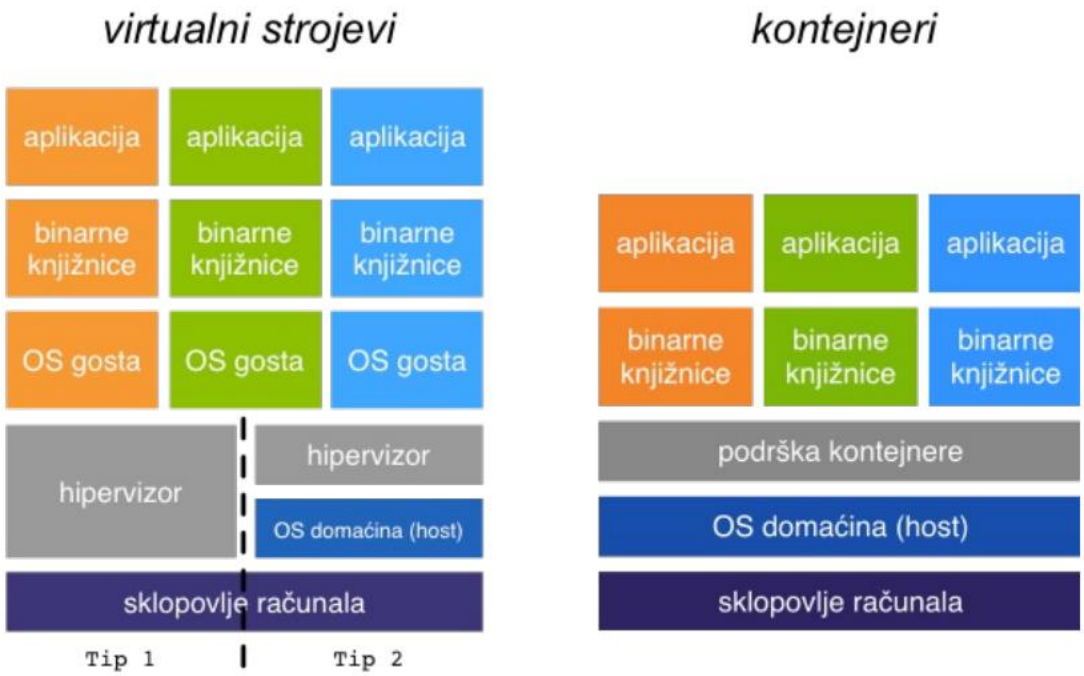
- Omogućuje izvršavanje više operacijskih sustava na jednom fizičkom ili na više fizičkih računala
- Isto vrijedi za pohranu podataka
- Resursi se mogu dijeliti ili udruživati.
- Računala mogu imati podršku za virtualizaciju (tehnologija hipervizora)
- Primjeri: Xen, VMware, Wine, …

Objasnite modele računarstva u oblaku.



Koja je razlika između virtualnih strojeva i kontejnera? Usporedite virtualne strojeve i kontejnere.

Kontejner je skup izoliranih procesa u korisničkom prostoru OS-a, koristi manje resursa i brže se pokreće od virtualnih strojeva koji moraju pokretati čitav OS. Zbog toga je u kontejneru teže kontrolirati korištenje resursa, izolacija između kontejnera je slaba i svaki kontejner mora koristiti kernel domaćinovog OS-a.



Što je to OCI specifikacija i što definira?

Open Container Initiative specification osnovana je 2015. godine na inicijativu Dockera i ostalih kompanija kako bi definirale specifikacije koje bi svaki kontejner trebao imati (lakša interoperabilnost između različitih alata koji rade s kontejnerima).
Definira:

- Izvršnu specifikaciju
- Specifikaciju slika
- Specifikaciju distribucije slika

Objasnite kako se Docker izvršava na različitim operacijskim sustavima (Linux, Windows, MacOS).

Docker se nativno izvršava na Linuxu, samo ga je potrebno dohvatiti putem terminala. Na Windows-u i MacOS-u potreban je virtualni stroj s Linuxom. Najpoznatije rješenje je Docker Desktop koji na MacOS-u koristi native Hypervisor.framework a na Windows-ima Microsoft Hyper-V kako bi se Docker mogao pokrenuti. Također postoji Docker Machine koji koristi VirtualBox za stvaranje stroja ali se danas rijetko koristi.

Što je to Kubernetes?

Sustav za orkestraciju kotejnera nastao u Google-u 2014. godine. Neovisan je o oblaku u kojem se izvršava, preko pružatelja usluga dostupan je kao PaaS ili IaaS. Korisnik definira krajnje stanje a k8s se brine za održavanje tog stanja.

Koji su osnovni koncepti koji se koriste u Kubernetesu?

node (čvor) – jedno virtualno računalo na kojem se izvršava k8s

pod (kapsula) – najmanji koncept koji se isporučuje, u sebi može imati više kontejnera

replica set – pravila za repliciranje, pazi da određeni broj replika imamo u sustavu

deployment – pravila za čitavu aplikaciju, brine se za verzije slika

service – kada se izvana pristupi usluzi onda se promet preusmjeri na neki od kapsula i na određena vrata (tj. kontejner unutar te kapsule)

namespace – odvaja dijelove k8s grozda može se definirati ograničenje korištenja resursa (quota), mogu se definirati prava korisnika i obično devops tim definira namespace i daje nam prava

label – oznaka nekog koncepta

selector – definira na koje koncepte se odnosi pojedini dio konfiguracije. Odabire se na temelju oznaka (label)

Što je i čemu služi Istio?

Služi za automatsko upravljanje opterećenjem za razne protokole (HTTP, gRPC, WebSocket i TCP). Omogućuje finu kontrolu prometa (pravila usmjeravanja, ponovno slanje zahtjeva,…), definiranje politika (pravo pristupa, ograničenja), automatske metrika, log-ova i trace-ova te omogućuje sigurnu komunikaciju između usluga sa zaštitom identiteta i kriptiranjem komunikacije.

Koji su elementi životnog ciklusa raspodijeljenog sustava?

Definicija zahtjeva, analiza rješenja, sinteza, ispitivanje, rad, mjerenja i modifikacija zahtjeva.

Koji su najvažniji nefunkcijski zahtjevi?

Performanse, raspoloživost i ukupna cijena vlasništva.

Kako se oni skupno zovu?

Kvaliteta usluge (QoS).

Što je to ugovor o razini usluge (SLA)?

Ugovor između korisnika i davatelja usluge koji definira razinu usluge.

Zašto je vrednovanje performansi važno?

Zbog planiranja kapaciteta koji je potreban za uspješan rad

Koje metode analize se koriste u praksi?

Iskustvo, modeliranje i simulacija

Kakve vrste modela tereta postoje u praksi?

Prirodne aplikacije, umjetne aplikacije (benchmarks) i neizvodivi modeli opisani intenzitetom zahtjeva, prosječnim vremenom obrade i sl.

Koji su koraci pri razvoju modela sustava?

Razumijevanje funkcioniranja, modeliranje tereta, mjerenje sustava u pogonu radi utvrđivanja parametara tereta, razvoj modela, verifikacija i validacija, analiza mogućih scenarija promjena, prognoza promjena tereta u budućnosti, prognoza performansi sustava nakon puštanja u pogon te u budućnosti

Koje su najčešće greške kod modeliranja?

Presložena analiza, nema specifičnog cilja, prejudiciranje, nedovoljno razumijevanje sustava, neadekvatne mjere nereprezentativni teret, neuključivanje važnih parametara, promatranje u

krivom intervalu vrijednosti parametara, krivo baratanje ekstremima, nedovoljno promatranje evolucije sustava i tereta, kriva interpretacija rezultata.

Što definiraju pojmovi MTBF i MTTR?

Srednje vrijeme između pogrešaka i srednje vrijeme popravka.

Kako je definirana raspoloživost sustava?

D = MTBF/(MTBF + MTTR)
Postotak vremena u kojemu je sustav na raspolaganju korisnicima.

Kako se računa raspoloživost paralelno (Dp) i serijskih (Ds) povezanih sustava?

Dp = (1-D1)*D2 + (1-D2)*D1 + D1D2
Ds = D1*D2

Koje su dvije osnovne grupe troškova za gradnju i pogon web-sustava i kako su obično raspodijeljeni?

Kapitalni i operativni troškovi, grubo raspodijeljeni 50/50 kroz tri godine.

Objasnite pojam vremena odziva i kapaciteta

Vrijeme odziva određuje odziv na jedan zahtjev.
Kapacitet odgovara maksimalnom broju zahtjeva koji se mogu obraditi u jedinici vremena.

Koje se metode za poboljšanje performansi koriste u arhitekturi raspodijeljenih sustava?

Serijsko preklapanje, paralelno preklapanje, paralelno izvođenje i privremena pohrana.

Kako je definirano ubrzanje?

Ubrzanje je omjer vremena izvođenja na jednom i više paralelnih podsustava.

Što su tipični uzroci natjecanja za sredstva (contention) ili potrebe za usklađivanjem podataka (coherence)?

- Zajedničke funkcije i varijable u operacijskom sustavu
- Izmjena zajedničkih podataka koji se mijenjaju u privremenim spremnicima (cache)
- Promet podataka u/iz glavne memorije
- Čekanje na ulaz/izlaz
- Sinkronizacijski primitivi

Koliko razina protokola uključuju web-aplikacije kojima se pristupa putem lokalnih mreža Ethernet spojenih na Internet?

Četiri: Ethernet, IP, TCP i HTTP

Što uključuje vrijeme odziva web-aplikacije?

Vrijeme pristupa klijenta (klijent – ISP), vrijeme prijenosa paketa od klijentovog do poslužiteljevog ISP-a (ISP – ISP), vrijeme dostupa do poslužitelja (ISP – poslužitelj), ISP – davatelj internetske usluge (Internet Service Provider).

Kako se može ostvariti razmjeran rast aplikacije?

Vertikalno ili horizontalno

Koje probleme donosi horizontalan rast?

Usklađivanje, raspodjela tereta i raspodjela podataka

9. Mikrousluge (mikroservisi), komunikacija temeljena na događajima

Koji su nedostaci monolitne arhitekture u odnosu na mikrousluge?

Monolitna arhitektura vezana je za jednu tehnologiju, tj. vezana je za jedan programski jezik, programske okvire i obično relacijske baze podataka. Nije moguće skalirati samo dio aplikacije već cijelu aplikaciju u također je potrebno shvaćanje rada cijele aplikacije. Svaka pojedinačna usluga mikrousluge može se zasebno isporučiti dok se kod monolitne arhitekture mora isporučiti čitava aplikacije.

Na koji način se radi komponentizacija monolita na mikrousluge?

Svaka usluga mora činiti jednu poslovnu funkcionalnost (Single Responsibility Principle). Potrebno je pažljivo definirati sučelja, ako se promijeni jedna mikrousluga ne smije se zahtijevati promjena drugih mikrousluga. Ako se su sučelja mijenjaju treba ih napraviti tako da promjene ne uzrokuju trenutnu promjenu kod drugih usluga.

Navedite 4 problema s kojima se suočavamo kada implementiramo mikrousluge.

Kako pratiti kompleksnost sustava?
Kako znati koji build, tj. koja verzija je na kojem čvoru?
Kako doći do dnevničkih zapisa (raspodijeljeni logging)?
Kako pratiti obradu jednog zahtjeva (traceability)?

Kako se rješava konfiguriranje velikog broja mikrousluga?

Konfiguracija se stavlja u neku datoteku kako bi se lakše mogla instalirati u drugoj okolini. Može se nalaziti u arhivi u kojoj se isporučuje, konfiguracijska datoteka, varijablama okoline ili ovisi o specifičnom rješenju korištenog oblaka.

Čemu služi registracija mikrousluga u registru?

Omogućuje otkrivanje usluga s poslužiteljske strane (komponenta za uravnoteženje brine o otkrivanju usluga) ili otkrivanje usluge s klijentske strane (klijent brine gdje se nalazi usluga).

Kako možemo raditi uravnoteženje opterećenja u sustavu s mikrouslugama?

Ako koristimo Kubernetes onda postoji posebna komponenta u grozdu koja radi uravnoteženje ravnomjerno s obzirom na broj instanci. Istio koristi proxy kod svake usluge kako bi puno finije mogao usmjeravati promet ovisno o opterećenju.

Čemu služi API Gateway?

Jedinstvena točka pristupa za klijente, prihvaća HTTP(s)/WebSocket zahtjeve i preusmjerava ih na jednu ili više mikrousluga. Moguće je i napraviti API Gateway za svaku vrstu klijenta (mobilni, web, javni pristup,…).

Navedite nekoliko metrika mikrousluga i tehnologije koje služe za njihovo praćenje.

Metrike mikrousluga: zahtjev/sekundi, memorija, procesor, vrijeme odgovora, probe (liveness, readiness).
Tehnologije za njihovo praćenje: Prometheus, Grafana.

Čemu služi agregiranje dnevničkih zapisa (log) i koje tehnologije koristimo za to?

Služi prikupljanju zapisa svih usluga na jedno mjesto, najčešće se koristi ELK stack:

- Elasticsearch – spremanje podataka, indeksiranje i pretraživanje
- Logstash – čita podatke iz datoteke/mreže, transformira ih i sprema u elasticsearch
- Kibana – vizualizacija podataka (grafovi, pite, ...) iz elasticsearcha

Koji problem rješava praćenje (distributed tracing) zahtjeva u sustavi s mikrouslugama?

Olakšava pronalaženja izvora greške ili kašnjenja unutar mikrousluge.

Objasnite način rada arhitekturnog obrasca osigurača.

Osigurava da se izolira dio koji ne radi i da se prestanu slati zahtjevi na taj dio, tako sprječava kaskadni ispad. Za tu funkcionalnost koristi sljedeća stanja:

- **Close** – svi pozivi se prosljeđuju na krajnju uslugu i vraćaju se odgovori kakvi se dobiju od krajnje usluge (uspješno, neuspješno, timeout). Računa se uspješnost poziva, ako padne ispod definirane granice prelazi se u stanje Open.
- **Open** – niti jedan poziv se ne prosljeđuje usluzi, na zahtjev se vraća neka podrazumijevana vrijednost. Postoji definirano vrijeme koliko se usluga nalazi u tom stanju a nakon tog vremena prelazi u stanje Half Open.
- **Half Open** – dio poziva se prosljeđuje usluzi, računa se uspješnost i ako je iznad definirane granice prelazi u stanje Close u suprotnom se vraća u stanje Open.

Navedite i objasnite strategije uvođenja novih verzija usluga.

Recreate
- ugasi staru verziju i pokreni novu

Ramped
- kada imamo više replika onda dodamo jednu novu verziju, pa ugasimo staru, i tako dok sve ne zamijenimo (Kubernetes)

Blue/Green
- instaliramo novu verziju (*blue*) i testiramo da radi, nakon toga samo sav promet preusmjerimo na novu verziju (*switch*)

Canary
- slično Blue/Green samo postepeno postotak sa stare verzije prebacujemo na novu (Istio)

A/B Testing
- samo neke korisnike prebacimo na novu verziju, primjeri kriterija: beta testeri, geografsko područje, korisnici pojedinog preglednika, jezik, OS, veličina ekrana, ... (Facebook, Istio)

Shadow (Dark Launch)
- instaliramo obje verzije i zahtjeve šaljemo na obje, ali samo od stare se odgovori vraćaju korisniku (testiranje u produkciji)

Navedite i objasnite verzioniranje poruka i njihovu kompatibilnost.

Prije slanja i primanja provjerava poruka se provjerava u registru kompatibilnosti sheme, postoje dvije vrste kompatibilnosti:

- **Prema natrag** – primatelj s novom verzijom može primiti stare verzije od pošiljatelja
- **Prema naprijed** – primatelj sa starom verzijom može primati nove verzije

Koji je problem raspodijeljenih transakcija i zašto se ne koriste kod mikrousluga?

?

Objasnite pojam eventualne konzistentnosti.

Sustav će nakon nekog vremena nakon zadnjeg zahtjeva doći u konzistentno stanje.

Objasnite arhitekturni obrazac CQRS (Command Query Responsibility Segregation).

?

Čemu služi raspodijeljena saga?

Kako bi se dugački zahtjevi mogli prekinuti i po potrebi promjene koje su se dogodile poništiti. Mora omogućiti isporuku zahtjeva najviše jednom a kompenzirajući zahtjev se mora isporučiti najmanje jednom (moraju biti idempotentni moraju se moći izvršiti).

Koja su svojstva računarstva na rubu?

Smanjuje se količina podataka koji se prenose mrežom jer se ne moraju svi podaci slati u oblak, povećana sigurnost podataka jer podaci ne izlaze iz lokalne mreže (GDPR), brže vrijeme odziva jer je manje kašnjenje u mreži.

10. Replikacija podataka

Objasnite vezu između replikacije i konzistentnosti.

Raspodijeljeni sustav je konzistentan u nekom vremenskom trenutku ukoliko su sve replike u njemu nalaze u istom stanju.

Objasnite način replikacije u HDFS-u.

- Svaki **DataNode** periodički šalje poruke **NameNode**-u
  - Porukama **HeartBeat** daje do znanja da je i dalje raspoloživ
  - Porukama **BlockReport** dojavljuje koji blokovi podataka su pohranjeni kod njega
- Na osnovu ovih poruka **NameNode**
  - Inicira dodatnu replikaciju ukoliko neki **DataNode**
    - Postane nedostupan (npr. zbog tehničkog kvara) ili
    - Izgubi blokove podataka pohranjene kod njega (npr. u slučaju problema na nekom od njegovih tvrdih diskova)
  - Inicira brisanje viška replika prilikom vraćanja u sustav privremeno nedostupnog **DataNode**-a
  - Prosljeđuje klijentske zahtjeve za pisanje i čitanje blokova podataka samo na ispravne čvorove **DataNode**

Navedite vrste replika i objasnite glavne razlike između njih.

Trajne replike – početni skup replika postavljen na skupu računala povezanih lokalnom mrežom, statička organizacija, većina zahtjeva je čitanje i raspoređivanje zahtjeva na dostupne replike.

Korisničke replike – dohvaćeni podaci spremaju se u lokalne spremnike i potrebno je održavati njihovu konzistentnost pomoću poslužitelja s kojeg su podaci dohvaćeni.

Poslužiteljske replike – sadrži trajne replike koje su dostupne korisnicima, poslužitelj prati vlastito stanje opterećenja, odabire i raspoređuje replike dinamički tijekom rada sustava.

U sustavu replika koji se sastoji od glavnog poslužitelja i n=6 podjednako opterećenih pomoćnih poslužitelja, izračunajte prosječno mrežno opterećenje glavnog poslužitelja za sljedeće metode održavanja konzistentnosti: a) pull, b) push s prosljeđivanjem novog sadržaja c) push s prosljeđivanjem operacija za promjenu sadržaja i d) push s prosljeđivanjem obavijesti o promjeni sadržaja. Pri tome pretpostavite da korisnike poslužuju samo pomoćni poslužitelji, da je prosječna frekvencija upita fu=500 upita/s, prosječna frekvencija promjena fp=4 promjene/min te da su prosječne veličine replika, upita/odgovora i operacija za promjenu sadržaja replika lr=800 kb, lp=10 kb i lo=600 kb.

?

Koja je razlika između aktivne i pasivne replikacije? Koje jedinice grupne komunikacije se koriste u jednoj i drugoj?

U aktivnoj replikaciji klijent komunicira s grupom replika koristeći TOCAST, zahtijeva isporuku poruka u istom redoslijedu svim replikama u skupini. U pasivnoj replikaciji klijent komunicira s glavnom replikom koristeći VSCAST, ako se glavna replika raspadne mora se izbrati nova glavna replika, zahtijeva da su poruke o promjenama ispravno uređene čak i nakon ispada glavne replike.

Koji je glavni problem aktivne replikacije i kako ga rješava polu-aktivna replikacija?

Glavni problem aktivne replikacije je determinizam provođenja operacija kod pojedine replike (redoslijed njihovog provođenja drugačiji je od redoslijeda prispijeća). Polu-aktivna replikacija rješava taj problem tako da glavna replikacija donosi odluku i obavještava ostale replike, kao kod pasivne replikacije.

Koja je sličnost između aktivne replikacije kod raspodijeljenih sustava i revne replikacije s promjenama bilo koje replike kod raspodijeljenih baza podataka?

?

Zašto je neophodno usuglašavanje kod lijene replikacije s promjenama bilo koje replike?

Koordinacija u slučaju s promjenama bilo koje replike (update everywhere) je problematična zbog mogućih konflikata u transakcijama, zato je potrebno usuglašavanje da se odredi koje transakcije trebaju provesti a koje odbaciti.

11. Particioniranje i konzistentnost podataka

Objasnite postiže li se particioniranjem uvijek bolja skalabilnost od korištenja replikacije.

Podaci su često indeksirani, a veći indeks najčešće ima bolje performance od particioniranog indeksa (tj. puno manjih indeksa particija), znači particioniranje ne more uvijek biti od skalabilnosti.

- Replicated Index and Subscriptions (RIS)
- Spatially Partitioned Subscriptions (sPS)
- Spatially Partitioned Index and Subscriptions (sPIS)
- Replicated Index Spatially Partitioned Subscriptions (RIsPS)
- Replicated Index Hash Partitioned Subscriptions (RIhPS)

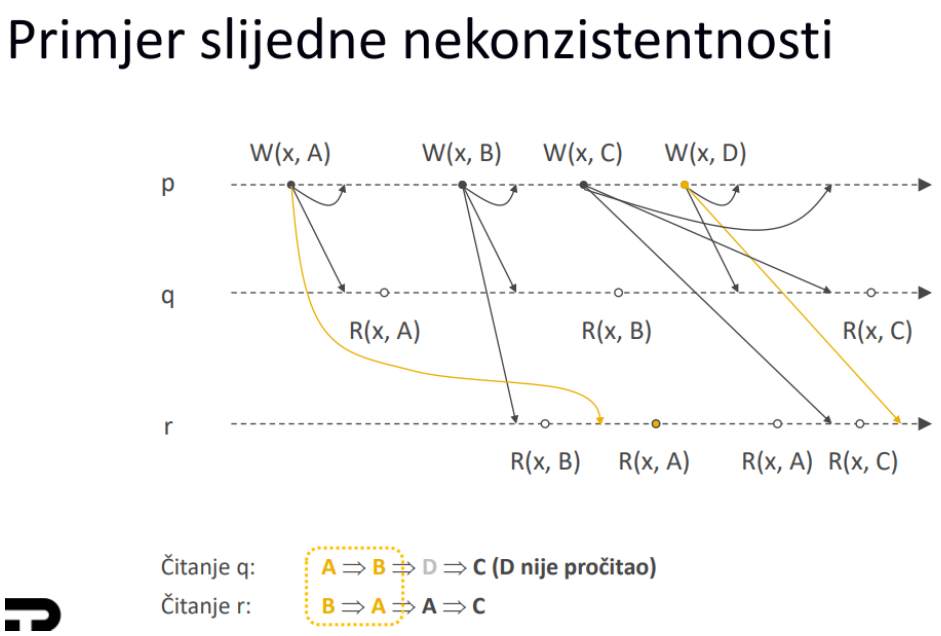
Objasnite asimetričnost particija na jednom primjeru.

Particije su nejednakih veličina, može se pojaviti tijekom duljeg rada raspodijeljenog sustava. Dovodi do nejednolikog opterećenja čvorova u sustavu i degradira performance čitavog raspodijeljenog sustava.

Koji je nedostatak particioniranja u krug odnosu na nasumično particioniranje?

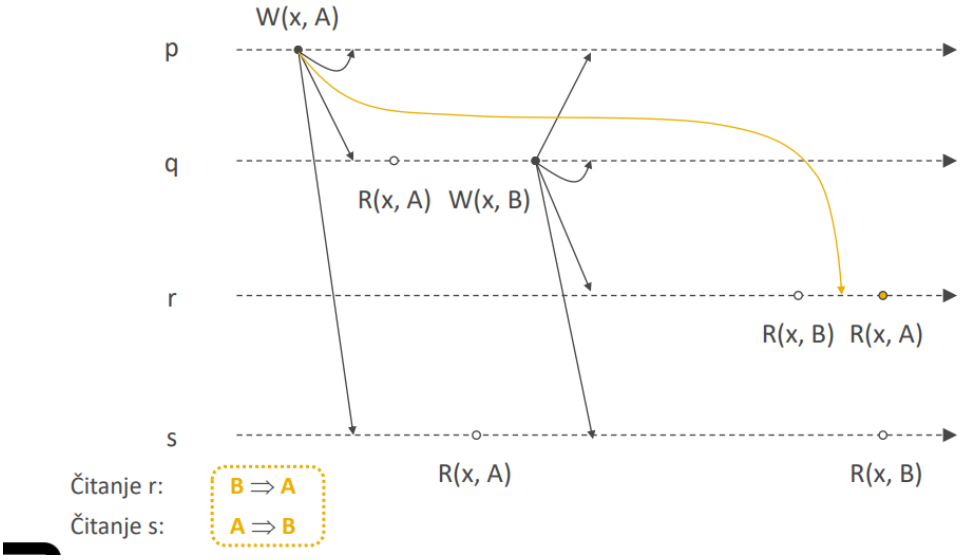
Kod particioniranja u krug potrebno je reparticioniranje svaki puta kada se promijeni broja particija.

Objasnite poštuje li se slijedna konzistentnost u slijedu izvođenja operacija prikazanom na slajdu 35.



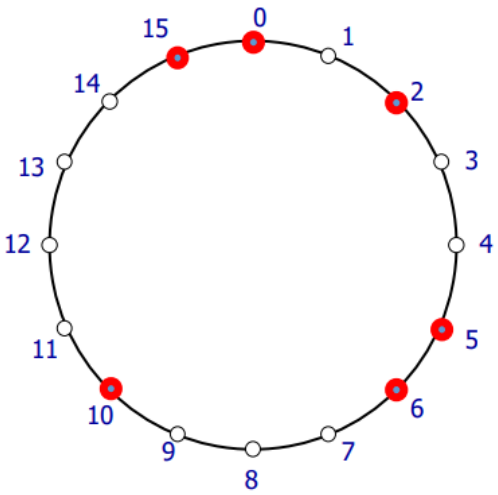
Primjer je za nekonzistentnost tako da ne poštuje se slijedna konzistentnost.

Objasnite poštuje li se povezana konzistentnost u slijedu izvođenja operacija prikazanom na slajdu 38. ukoliko pretpostavimo da je operacija pisanja podatka B (uzročno) povezana s operacijom pisanja podatka A.



Primjer je za nekonzistentnost tako da ne poštuje se povezana konzistentnost.

12. Sustavi s ravnopravnim sudionicima



Za Chordov prsten sa slike navedite tablicu usmjeravanja za čvor 5.

Prsten ima 16 čvorova, N = 16, tako da je m = 4. Tablica se zatim računa ovako:

Indeks	Čvor susjed
$5 + 2^0 = 6$	6
$5 + 2^1 = 7$	10
$5 + 2^2 = 9$	10
$5 + 2^3 = 13$	15

Navedite korake koji će pohraniti podatak d za koji vrijedi H(d)=13 ako podatak želi pohraniti čvor čiji je ključ jednak 2. Na kome čvoru će podatak biti pohranjen?

Upit kreće iz čvora 2, u svojoj tablici gle koji mu je susjed najbliži čvoru 13 (budući da čvor 13 ne postoji gleda se prvi najbliži a to je čvor 15). Na indeksu $2 + 2^3 = 10$, a čvor 10 na indeksu $10 + 2^2$ ima susjeda čvor 15 na kojeg se zatim posprema podatak d.

Čvor 2 -> Čvor 10 -> Čvor 15

Navedite korake za pronalaženje podatka d iz prethodnog pitanja ako upit za d dolazi s čvora s ključem 0. Koja je cijena u smislu komunikacije među peerovima?

Upit za traženje kreće iz čvora 0, budući da svaki čvor zna na temelju podataka ključ gdje se nalazi podatak čvor 0 zna da mora doći do čvora 15. Na indeksu $0 + 2^3$ pronalazi čvor 10, a čvor 10 na indeksu $10 + 2^2$ ima upisan čvor 15 na kojem se nalazi podataka d.

Čvor 0 -> Čvor 10 -> Čvor 15

13. Tehnologija raspodijeljene glavne knjige

Nema pitanja za ponavljanje