Pojmovi

- **Distribuirano izvršavanje** je kada se dva posla izvršavaju u fizički razdvojenim adresnim prostorima
  - ranije je korišćeno "na fizički razdvojenim računarima" ali danas se smatra da izvršavanje na različitim računarima, a koji dele radnu memoriju, nije distribuirano izvršavanje
  - alternativno "...kada dva posla mogu da komuniciraju isključivo posredstvom računarske mreže"

Pojmovi

- **CBP – Centralizovana baza podataka** je baza podataka koja u celosti počiva na jednom računaru
- **CSUBP – Centralizovani sistem za upravljanje bazama podataka** je softverski sistem koji omogućava upravljanje CBP-om

Pojmovi

- **DBP - Distribuirana baza podataka** je skup više **logički međuzavisnih** baza podataka koje su distribuirane na računarskoj mreži
  - tj. postoje pravila integriteta koja se odnose na više baza podataka
- **DSUBP - Distribuirani sistem za upravljanje bazama podataka** (DDBMS – distributed DBMS) je softverski sistem koji omogućava upravljanje DBP tako da je **distribuiranost transparentna za korisnika**

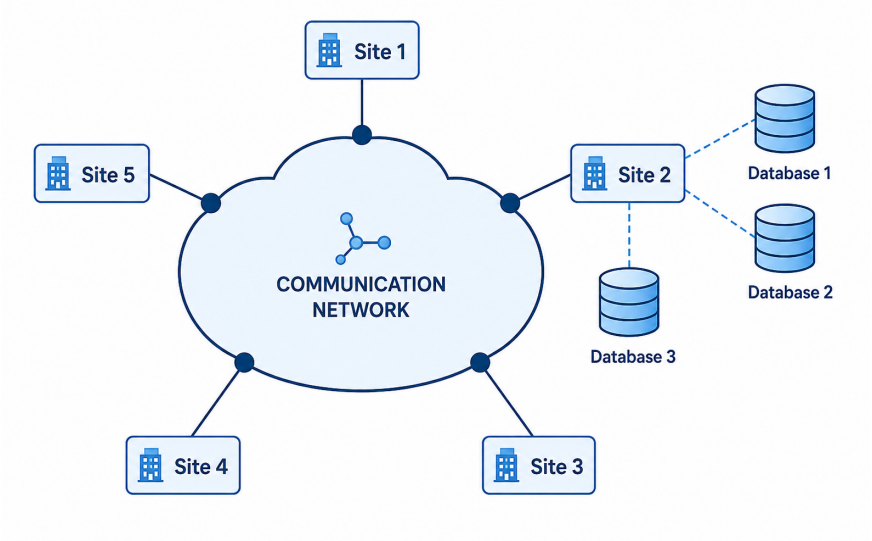
Distribuiranost i baze

- U kontekstu razmatranja distribuiranih baza podataka koristi se isključivo stroga definicija distribuiranosti
  - suštinski je važno da su distribuirane komponente na različitim računarima
    - ali udaljenost nije značajna, mogu da budu jedan do drugog ili na različitim kontinentima
- Pretpostavlja se da je jedini deljeni resurs **mreža**

Baze podataka na mreži

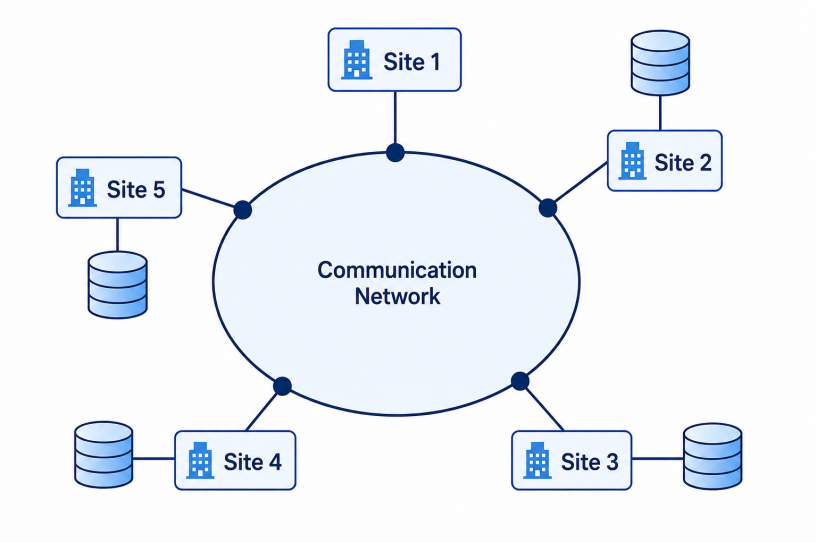
- Činjenica da se SUBP nalazi u okviru mreže nije dovoljna da bi se radilo o DSUBP
- Ako je na računarskoj mreži sa većim brojem računara baza podataka locirana *na samo jednom čvoru*, tada to nije DSUBP
- SUBP je distribuiran akko je **lociran na više različitih čvorova** u mreži

Baze podataka na mreži



Nije DSUBP

Baze podataka na mreži



Jeste DSUBP

Predmet distribuiranja

- Ključno pitanje je **šta** se distribuira?
- Može da se distribuira praktično sve:
  - logika obrade
  - pojeidnačne funkcije
  - podaci
  - upravljanje obradom ili transakcijama

Predmet distribuiranja (2)

- **Distribuiranje logike obrade**
  - obrada se distribuira među jedinicama za obradu (čvorovima)
  - podrazumeva visok nivo konceptualne homogenosti
    - svi delovi distribuiranog sistema imaju isto viđenje podataka, operacija i načina rada sistema
    - iako hardver ne mora biti isti, softver najčešće mora
- **Distribuiranje pojedinačnih funkcija**
  - različite funkcije se izvršavaju na različitim čvorovima
  - neke (ili sve) funkcije mogu da se izvršavaju na više čvorova (replikacija funkcija)
  - nije neophodan visok nivo homogenosti
    - dovoljno je da podsistemi mogu da komuniciraju, nije neophodna homogenost ni na nivou hardvera ni na nivou softvera

Predmet distribuiranja (3)

- **Distribuiranje podataka**
  - podaci mogu da se nalaze na većem broju čvorova
    - horizontalna i vertikalna fragmentacija
    - replikacija podataka
  - katalog podataka može da bude centralizovan, repliciran ili distribuiran
- **Distribuiranje upravljanja**
  - neki ili svi poslovi upravljanja mogu biti distribuirani
    - upravljanje transakcijama
    - upravljanje privilegijama
    - upravljanje obradom, tj. organizacijom izvršavanja obrade

Kriterijumi za posmatranje

- Distribuirani računarski sistemi mogu da se posmatraju i klasifikuju na osnovu različitih kriterijuma:
  - stepen međusobne spregnutosti čvorova
  - struktura međusobne povezanosti
  - međuzavisnost komponenti
  - sinhronizacija među komponentama

Stepen spregnutosti

- Stepen spregnutosti je mera koja opisuje koliko su čvrsto međusobno povezani elementi sistema
  - Može da se opiše numerički, na različite načine
    - relativna mera, nema mnogo smisla pri poređenju različitih sistema
  - Češće se opisuje rečima, kao *niska* ili *visoka* spregnutost
    - niska spregnutost elemenata označava da oni nemaju deljenih podisistema i komuniciraju putem računarske mreže, dok
    - visoka spregnutost elemenata označava da oni imaju neke deljene podsisteme (npr. memoriju, diskove ili posvećene komunikacione linije) i da komuniciraju na neki efikasniji način

Stepen spregnutosti (2)

- Primer numeričkog opisivanja stepena spregnutosti
  - *Cpl = AvgDataExchg / AvgLocalProcess*
    - *AvgDataExchg* = prosečna količina podataka koji se razmenjuju između nekog čvora sistema sa ostalim čvorovima u jedinici vremena
    - *AvgLocalProcess* = prosečna količina posla obavljenog na tom čvoru u jedinici vremena
  - na primer:
    - ako jedan čvor pri prosečnom opterećenju u jedinici vremena razmeni sa drugima u proseku 250MB
    - ...i pri tome izvrši 550MI (miliona instrukcija)
    - ...onda je njegova spregnutost 0,40
  - visoka spregnutost može da označava nisku samostalnost čvora u obavljanju poslova

Struktura međusobne povezanosti

- Razlikuju se dva osnovna slučaja:
  - neposredno povezivanje čvorova (P2P – peer to peer) putem sopstvenih rezervisanih kanala veze
  - upotreba deljenih kanala veze za komunikaciju među svim sistemima
- U praksi su česte kombinovane složene strukture

Međuzavisnost komponenti

- Međuzavisnost komponenti opisuje u kojoj meri komponente zavise jedna od druge pri radu
- Primer, prema funkcijama:
  - visoka – ako komponenta A ne može da završi posao bez pomoći komponente B u realnom vremenu
  - niska – ako otkaz komponente B neće značajno ometati rad komponente A

Sinhronizacija među komponentama

- Komponente mogu da komuniciraju
  - sinhrono i
  - asinhrono
- Ako je komunikacija sinhrona, međuzavisnost je visoka
  - uspeh obavljanja posla zavisi od uspeha komunikacije i izvršavanja odgovarajućih poslova na drugom čvoru
- Ako je komunikacija asinhrona, međuzavisnost nije obavezno niska

Doprinosi DSUBP

- Osnovni doprinosi DSUBP su:
  - Transparentno upravljanje distribuiranim i repliciranim podacima
  - Pouzdanost distribuiranih transakcija
  - Unapređenje performansi
  - Lakše proširivanje sistema

(pod *doprinosima* se misli na naša očekivanja, tj. na ono što će DSUBP da nam dodatno pruži, u odnosu na prost skup više CSUBP)

Transparentno upravljanje distribuiranim i repliciranim podacima

- Transparentnost upravljanja podrazumeva da se DSUBP bavi razdvajanjem semantike visokog nivoa od problema koji nastaju pri implementaciji na niskom nivou
- Na primer:
  - neka su u bazi podataka banke pohranjeni podaci o zaposlenima u svim ekspoziturama
  - nezavisno od toga gde se fizički čuvaju podaci, da li su distribuirani ili ne i slično, podacima mora da se pristupa na isti univerzalan način (na primer, putem upita na upitnom jeziku)
  - svu složenost koja nastaje kao posledica distribuiranja rešava DSUBP i sakriva je od korisnika

Transparentno upravljanje distribuiranim i repliciranim podacima (2)

- Transparentnost upravljanja podacima ima sledeće aspekte:
  - Nezavisnost podataka
  - Mrežna transparentnost
  - Transparentnost replikacije
  - Transparentnost fragmentacije

Transparentno upravljanje distribuiranim i repliciranim podacima (3)

- **Nezavisnost podataka**
  - Logička nezavisnost podataka
    - Otpornost aplikacije na promene logičke strukture podataka
    - Aplikacije dobro podnose dodavanje novih elemenata strukturi baze podataka
  - Fizička nezavisnost podataka
    - Potpuno skrivanje fizičke strukture podataka od aplikacije
    - Aplikacija ne sme da trpi nikakve posledice u slučaju promene fizičke strukture podataka, makar ona uključivala i promene u načinu distribuiranja podataka

Transparentno upravljanje distribuiranim i repliciranim podacima (4)

- **Mrežna transparentnost**
  - U distribuiranim sistemima, mrežna struktura je potrebno da bude sklonjena od očiju korisnika i transparentno upravljana
    - Korisniku nije potrebno da zna gde su podaci i kako čvorovi komuniciraju
  - Naziva se i **distributivna transparentnost**

Transparentno upravljanje distribuiranim i repliciranim podacima (5)

- **Transparentnost replikacije**
  - Replikacija je čuvanje istih podataka na više lokacija
    - kao i obavljanje istih poslova na različitim čvorovima
    - preduzima se radi performansi, raspoloživosti i pouzdanosti
  - Korisnici ne bi trebalo da budu svesni činjenice da se podaci i poslovi repliciraju
    - bilo da su replicirani ili ne, oni se moraju koristiti na isti način

Transparentno upravljanje distribuiranim i repliciranim podacima (6)

- **Transparentnost fragmentacije**
  - Fragmentacija je čuvanje različitih delova iste kolekcije podataka na više lokacija
    - Preduzima se radi performansi, raspoloživosti i pouzdanosti
  - Korisnici ne bi trebalo da budu svesni činjenice da su podaci fragmentisani
    - Bilo da su fragmentisani ili ne, oni moraju da se koriste na isti način
    - Obično se upiti (tzv. globalni upiti) prevode u izvršavanje kolekcije manjih (tzv. lokalnih upita) i rezultati se uniraju

Pouzdanost distribuiranih transakcija

- Nosilac pouzdanosti u radu SUBP-a je **transakcija**
- Izvođenje transakcija u distribuiranom okruženju donosi nove probleme
  - Npr. prevazilaženje otkazivanja jednog od udaljenih uređaja u fazi potvrđivanja transakcije

Unapređenje performansi

Doprinos DBP performansama obično se ogleda u tome što:

- Fragmentisani podaci mogu da se čuvaju bliže mestu upotrebe
  - Ravnomerno je raspodeljeno opterećenje na više servera na različitim lokacijama
  - Manje se vremena troši na prenos podataka
    - Iako može da izgleda da brze mreže umanjuju značaj ovog problema, činjenica je da se podaci distribuirano proizvode i koriste, pa će distribuirano čuvanje uvek biti prirodna organizacija
    - Posebno, brza mreža može da značajno smanji trajanje prenosa, ali ne tako dobro i početno kašnjenje odgovora
- Globalni upiti se paralelizuju
  - Podaci su distribuirani, pa se globalni upiti dele na manje, koji se odnose na lokalne podatke

Lakše proširivanje sistema

- Mnogo je lakše da se poveća kapacitet distribuiranog SUBP nego centralizovanog
  - skoro uvek je jevtinije da se ista snaga obezbedi pomoću više manjih računara nego pomoću jednog velikog
  - naravno da postoji prostor za velike računare, ali je u velikom broju slučajeva iz ekonomskog ugla opravdanije ulaganje u distribuirani sistem
    - manje inicijalno ulaganje u hardver
    - postepeno proširivanje u skladu sa realnim potrebama
    - može da predstavlja problem složeniji softver

Otežavajući faktori primene DSUBP

- Najvažniji otežavajući faktori su:
  - Složenost (pre svega softvera)
  - Cena
  - Distribuirana kontrola
  - Bezbednost

Otežavajući faktori

- **Složenost**
  - Distribuirani sistemi su sve samo ne jednostavni
  - Distribuirani SUBP mora da reši sve probleme koje rešava i centralizovan SUBP i još mnoge druge
- **Cena**
  - Distribuirani sistemi zahtevaju dodatni hardver (npr. komunikacioni)
  - Softverski aspekti sistema su mnogo složeniji i skuplji za razvijanje
  - Na svakoj lokaciji gde postoje serveri potrebno je i održavanje

Otežavajući faktori

- **Distribuirana kontrola**
  - Otežani su sinhronizacija i koordinacija komponenti
- **Bezbednost**
  - Kod centralizovanih BP staranje o bezbednosti je centralizovano
  - Ovde je distribuirano i uključuje dodatne aspekte
    - bezbednosti računarskih mreža
    - usklađivanja bezbednosnih aspekata na različitim čvorovima
    - povećan broj rizičnih ulaznih tačaka
    - i drugo

Problemi i teme DSUBP

- Najvažniji novi problemi koji se rešavaju pri upotrebi DSUBP su:
  - Projektovanje distribuiranih baza podataka
  - Distribuirano izvršavanje upita
  - Distribuirano upravljanje metapodacima
  - Distribuirana kontrola konkurentnosti
  - Distribuirano upravljanje mrtvim petljama
  - Pouzdanost distribuiranih SUBP
  - Podrška u operativnim sistemima
  - Heterogene baze podataka

Projektovanje distribuiranih BP

Novi problemi u kontekstu distribuiranja:

- Podaci mogu da budu
  - **particionisani** – (fragmentisani) podeljeni na više lokacija
  - **replicirani** – ponovljeni na više lokacija
- Moguće su i kombinacije
- U slučaju replikacije može se primeniti
  - **puna replikacija**
  - **delimična replikacija**
- U slučaju particionisanja
  - određuje se način fragmentisanja podataka
    - **horizontalna fragmentacija**
    - **vertikalna fragmentacija**
  - optimizuje se distribucija fragmenata

Distribuirano izvršavanje upita

- Izvršavanje upita obuhvata analizu upita i prevođenje upita u niz operacija
- Optimizacija upita je složeno pitanje i u centralizovanim BP, a u slučaju DBP to postaje još izraženije
- Dodatni faktori u odlučivanju su
  - cena mrežnog transporta
  - različitost cena izvršavanja na različitim lokacijama
  - različitost raspoloživih resursa na različitim lokacijama
  - mogućnosti paralelizacije

Distribuirano upravljanje metapodacima

- Metapodaci obuhvataju sve informacije koje se o strukturi, distribuiranosti i sadržaju baze podataka moraju čuvati da bi njen rad bio moguć
- Za razliku od centralizovanih BP, sada:
  - postoje dodatne informacije o lokacijama čvorova
  - postoje dodatne informacije o raspoređenosti podataka po čvorovima
    - fragmentacija
    - replikacija
  - svaki čvor mora da zna mnogo toga o drugim čvorovima, ali ne uvek sve
  - metapodaci mogu biti
    - centralizovani
    - replicirani
    - particionisani

Distribuirana kontrola konkurentnosti

- Kontrola konkurentnosti podrazumeva sinhronizaciju pristupanja podacima od strane konkurentno izvršavanih procesa
- Obuhvata
  - staranje o integritetu baze podataka (kao i za CBP)
  - staranje o konzistentnosti replika
    - **uzajamna konzistentnost**
  - složenije staranje o izolovanosti
    - ako se u transakciji menja neki podatak, onda moraju da se izoluju sve njegove replike
- Nekoliko pristupa
  - **pesimistički pristup**
    - sinhronizacija pre ostvarenog pristupa
  - **optimistički pristup**
    - sinhronizacija posle ostvarenog pristupa
- Koncepti
  - **zaključavanje** – počivaju na međusobnom isključivanju
  - **vremenski pečati** – počivaju na održavanju redosleda izvršavanja
  - **hibridni mehanizmi**

Distribuirano upravljanje mrtvim petljama

- Potencijalno viša složenost problema i konkurentnih odnosa nego kod CBP
- Rešava se na sličan način kao i na nivou operativnih sistema:
  - prevencijom
  - izbegavanjem
  - prepoznavanjem i oporavkom

Pouzdanost distribuiranih SUBP

- Pouzdanost ne dolazi sama po sebi, tj. za svaki od narednih doprinosa su potrebni neki preduslovi
  - Ako jedan čvor otkaže, ostali i dalje rade
  - Ako jedan čvor otkaže, drugi mogu da preuzmu njegovu ulogu
  - Kada se čvor oporavi, automatski se osvežava njegov sadržaj na osnovu drugih čvorova

Podrška u operativnim sistemima

- Operativni sistemi ne pružaju uvek odgovarajuće usluge složenim sistemima, kakav DSUBP izvesno jeste
- Problemi i zagušenja obično nastaju po pitanju
  - upravljanja radom procesora i raspodelom vremena
  - upravljanje procesima
  - upravljanja memorijom
  - rada sa fajlovima
  - oporavak
- Često se neki elementi implementiraju *ispod* nivoa operativnog sistema
  - npr. neposredan pristup hardveru
  - diskovi, memorijski moduli,...

Heterogene baze podataka

- DBP često nastaje spajanjem više postojećih rešenja
- Tada je obično narušena
  - homogenost softvera – jer različite baze rade pod različitim SUBP
  - homogenost strukture – jer struktura različitih baza podataka nije usaglašena
- I tada je potrebno rešavati neke od opisanih problema
  - distribuirano izvršavanje upita
  - upravljanje metapodacima
  - upravljanje konkurentnošću
- Ovakvi sistemi se često nazivaju **sistemi sa više baza podataka**
  - često se ne svrstavaju u DBP, tj. nisu prave DBP

Modeli arhitekture DSUBP

- Pri razmatranju potencijalnih arhitektura od značaja su tri osnovne dimenzije problema:
  - autonomija čvorova
  - distribuiranost čvorova
  - heterogenost čvorova
- Ove dimenzije su u najvećoj meri međusobno ortogonalne

Autonomija čvorova

- Autonomija čvorova se odnosi prvenstveno na upravljanje, a ne na skladištenje podataka
  - opisuje sposobnost čvorova da samostalno izvršavaju operacije
- Autonomija zavisi od velikog broja činilaca
  - da li i kako čvorovi razmenjuju informacije
  - da li čvorovi mogu nezavisno da izvršavaju transakcije
  - da li jedan čvor može menjati druge čvorove
  - da li jedan čvor može samostalno da upravlja lokalno ograničenim transakcijama
- Zahtevi mogu da se formulišu na različite načine
  - razmotrićemo autonomiju prema intenztetu povezivanja

Autonomija čvorova (2)

- Po intenzitetu povezivanja (Ozsu, Valduriez , 1999):
  - Tesna integracija
  - Poluautonomni sistemi
  - Puna izolovanost

Autonomija čvorova (2)

- **Tesna integracija**
  - jedna jedinstvena slika baze podataka je dostupna svim korisnicima svih čvorova
  - iz ugla korisnika, baza podataka kao da je centralizovana
  - svakim zahtevom korisnika upravlja tačno jedan od upravljača podacima, čak i kada njegovo izvršavanje izvodi više njih
  - upravljači podacima se uobičajeno ne ponašaju kao da su nezavisni sistemi, čak iako su obično sposobni za to

Autonomija čvorova (3)

- **Poluautonomni sistemi**
  - čvorovi sistema mogu da rade i obično rade nezavisno
  - uspostavljaju federaciju radi deljenja svojih lokalnih podataka
  - svaki čvor određuje koje će podatke staviti na raspolaganje drugim čvorovima
  - nisu potpuno autonomni zato što mogu da razmenjuju podatke

Autonomija čvorova (4)

- **Puna izolovanost**
  - svaki čvor je samostalan SUBP koji niti je svestan postojanja drugih SUBP niti komunicira sa njima
    - iz ugla čvora, svejedno je da li mu je neki zahtev uputio klijent (aplikacija) ili drugi čvor
  - izvršavanje globalnih transakcija je posebno složeno jer ne postoji puna globalna kontrola

Distribuiranost čvorova

- Distribuiranost se odnosi na podatke
  - razmatra se fizička distribuiranost podataka
  - korisnik sve podatke vidi kao jedinstvenu kolekciju
- Postoje tri osnovna nivoa distribuiranja
  - Centralizovani podaci (nema distribuiranja)
  - Klijent-server
    - (u ovom kontekstu klijent može da bude i drugi čvor)
  - Ravnopravni čvorovi

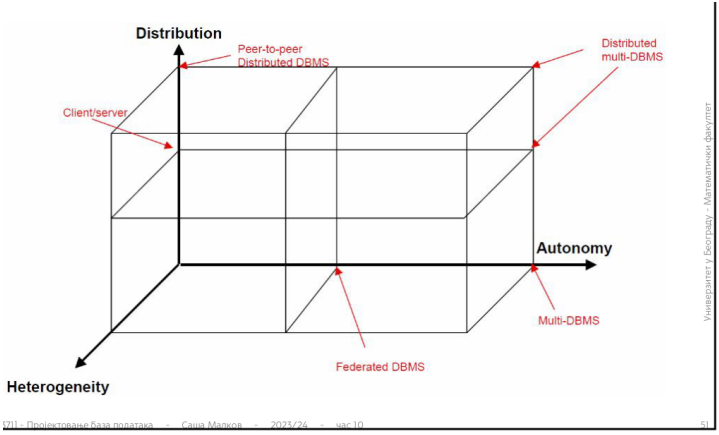
Heterogenost

- Heterogenost ima više aspekata i odnosi se
  - na hardver
  - na operativne sisteme
  - na mrežne protokole
  - na softver SUBP
- Posebno značajni aspekti su
  - modeli podataka
  - upitni jezici
  - protokoli upravljanja transakcijama

Osnovne alternative arhitektura

- Za svaku dimenziju uvodimo značajne tačke:
  - Autonomija:
  - 0 – tesna integracija
  - 1 – poluautonomni sistemi
  - 2 – puna izolovanost
  - Distribuiranost:
  - 0 – nema distribuiranosti
  - 1 – klijent/server sistemi
  - 2 – ravnopravni čvorovi
  - Heterogenost:
  - 0 – homogeni sistemi
  - 1 – heterogeni sistemi (delimično ili potpuno heterogeni)

Dijagram dimenzija arhitekture



Svojstva (uslovi) CAP

- Pri pravljenju distribuiranih sistema (ne samo baza podataka) kao najvažniji ciljevi se ističu tri važna svojstva:
  - konzistentnost (consistency)
  - raspoloživost (availability)
  - prihvatanje razdvojenosti (partition tolerance)
- Ova tri svojstva se skraćeno označavaju kao ***CAP***

Konzistentnost

- "Konzistentan sistem funkcioniše kao celina ili ne funkcioniše uopšte"
- "Sva čitanja, na svim čvorovima, moraju da daju isti rezultat"
- "**Rezultat operacije (čitanja) *nikada ne zavisi* od čvora na kome se izvršava**"
- Razlikuje se od istog termina u kontekstu osobina transakcija (A**C**ID)
- "... bez obzira na bilo kakva dešavanja sa bazom ili sistemom, ako sistem radi onda uvek mora da vrati ispravan rezultat..."

Raspoloživost

- "**Sistem je *uvek* raspoloživ**"
  - raspoloživost se praktično definiše kao odzivnost sistema u nekim garantovanim granicama
  - (u ovom kontekstu se razmatra samo vreme odziva)
- Paradoks je da sistem obično nije raspoloživ upravo onda kada je najpotrebniji
  - u vreme kada je raspoloživost najpotrebnija, onda je i najteže ostvariva, zato što je sistem tada obično najviše opterećen
  - ako je sistem raspoloživ kada nije potreban, to nema značaja

Tolerancija razdvojenosti

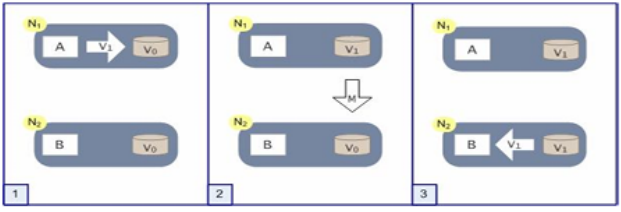
- "**Nijedan skup problema, osim potpunog otkazivanja, ne sme da proizvede neispravan odziv sistema**"
  - tj. sistem mora da prihvata delimične otkaze komunikacije i da nastavlja ispravno funkcionisanje
  - (u ovom kontekstu se razmatra ispravnost odziva)
- Povremeni prekidi komunikacije među čvorovima su neizbežni
- Razdvojenost (partition) je stanje komunikacione mreže u kome su delovi sistema (obično usled kvara) podeljeni na particije (dva ili više razdvojenih skupova čvorova) između kojih ne postoji komunikacija
- Očekuje se da sistem funkcioniše i daje ispravne rezultate čak i u uslovima razdvojenosti

Hipoteza CAP

- Eric Brewer, Berkley (2000)
- "Nije moguće definisati sistem koji zadovoljava sve CAP uslove (konzistentnost, raspoloživost i toleranciju razdvojenosti)."
- Moguće je definisati sistem koji zadovoljava izabrana dva od ovih uslova.

Teorema CAP (2)

- Pretpostavke:
  - neka imamo dva čvora i na njima repliciran podatak V:
    1. neka na čvoru 1 transakcija A ažurira V
    2. ažuriranje se porukom M propagira na čvor 2
    3. neka na čvoru 2 nešto kasnije transakcija B čita V
  - na slici je predstavljeno željeno ponašanje



Teorema CAP (3)

- Pretpostavimo da poruka M nije stigla na odredište
  - zbog *razdvojenosti* delova sistema
- U osnovi imamo tri moguća ponašanja sistema:
  1. transakcija A se poništava
     - to znači da sistem ***ne prihvata razdvojenost*** svojih delova, zato što zbog razdvojenosti onemogućava operacije ažuriranja
  2. transakcija A se uspešno nastavlja (i zatim završava)
     - to znači da sistem ***nije konzistentan***, zato što će čitanje podatka V na čvorovima 1 i 2 davati različite rezultate
  3. transakcija A čeka na uspešno slanje poruke M
     - to znači da sistem ***nije raspoloživ*** (nepoznato vreme odziva), zato što ne možemo da garantujemo vreme u kome će se transakcija završiti
- U svakom od slučajeva nije ispunjen po jedan od uslova

Kompromisi

- Zbog prethodnog problema, pri projektovanju (ili konfigurisanju) distribuiranog sistema neophodno je napraviti neki kompromis:
  - odbacivanje tolerancije razdvojenosti
  - odbacivanje raspoloživosti
  - odbacivanje konzistentnosti
  - ublažavanje uslova (tj. odbacivanje više strogo definisanih uslova)
  - zasnivanje sistema na drugačijem skupu uslova

Kompromisi (2)

- **Odbacivanje tolerancije razdvojenosti**
  - "pristajemo da sistem ne radi u slučaju razdvojenosti"
  - jedan način prevazilaženja je da sve bude na jednoj mašini
    - ali ako već moramo da pravimo DBP onda to verovatno nije prihvatljivo
  - alternativa je da se razdvojenost (a time i otkaz sistema) svede na najmanju moguću meru pomoću višestrukog umrežavanja
  - obe alternative su veoma skupe
- Imajući u vidu da se distriburana rešenja koriste samo onda kada su neophodna zbog obima podataka i poslova, ovaj vid kompromisa se retko pravi

Kompromisi (3)

- **Odbacivanje raspoloživosti**
  - "u slučaju razdvojenosti ne garantuje se vreme odziva"
  - posledice problema se umanjuju pažljivim projektovanjem sistema, tj. uspostavljanjem što niže sprege među čvorovima
- Sistem koji nije raspoloživ je praktično neupotrebljiv
  - zbog toga se ovaj pristup koristi relativno retko
    - ako je konzistentnost primarna
    - ako je tolerancija razdvojenosti nezaobilazna
    - teži se niskoj spregnutosti među čvorovima

Kompromisi (4)

- **Odbacivanje konzistentnosti**
  - "dopuštamo da isti upit daje različite rezultate na različitim čvorovima"
  - ako su razlike prihvatljivije nego niska raspoloživost
  - ako je učestalost pojavljivanja svedena na prihvatljivu meru
  - tj. ako je cena nekonzistentnosti prihvatljiva
- Konzistentnost je jedan od osnovnih uslova za uspešan rad baza podataka
- Ipak, postoje slučajevi kada nije primarna
  - npr. veb pretraživači
    - ako se nešto ne pronalazi na svakom čvoru, to nije veliki problem
  - čak i prodavnice
    - ako se mali broj proizvoda proda po staroj ceni, neće propasti firma

Izmenjen skup uslova - BASE

- Pojmovi iz teoreme počivaju na uobičajenim konceptima rada sa bazama podataka, tj. postoji zavisnost sa osobinama transakcija – ACID
- Ali, može da se definiše i drugačiji skup uslova, manje oštar, u kom slučaju sve odgovarajuće osobine mogu da se usklade
  - BASE umesto ACID

Izmenjen skup uslova - BASE (2)

- ***BASE***
  - ***B**asically **A**vailable*
    - ne garantuje se raspoloživost odgovora, već samo sistema
    - ako ne može da se dobije odgovor, bar će se dobiti obaveštenje o tome
    - i pored toga, visoka raspoloživost na štetu preostalih uslova
  - ***S**oft-state*
    - stanje sistema može da se menja čak i kada nije u toku nijedna transakcija (pre svega radi asinhronog ostvarivanja konzistentnosti)
  - ***E**ventually consistent*
    - žrtvuju se garantovana stalna konzistentnost i izolovanost transakcija zarad raspoloživosti
    - sistem će *u nekom trenutku (eventually)* postati konzistentan
    - ali će i pre tog trenutka raditi i davati odgovore (iako potencijalno različite na različitim čvorovima)

Izmenjen skup uslova - BASE (3)

- Suština koncepta je u **odloženom usklađivanju** sadržaja na različitim čvorovima
  - konzistentnost se ostvaruje, ali ne obavezno u okviru iste transakcije (eventual consistency)
  - sadržaj čvorova se menja i van odvijanja transakcija, a u cilju njihovog usklađivanja (soft state)
  - ako sistem nije u potpunosti raspoloživ, čvor može da pokuša da pruži manje pouzdan ili samo delimičan odgovor, uz odgovarajuću informaciju o tome (basically available)

Izmenjen skup uslova - BASE (4)

- Osnovna razlika je u trenutku usklađivanja podataka između čvorova, tj. u načinu obavljanja replikacije podataka
  - Kod ACID uslova
    - replikacija bi trebalo da se odvija u okviru transakcije
  - Kod BASE uslova
    - replikacije se odvija asinhrono, u nekom trenutku po završetku transakcije
    - tj. transakcija se odvija lokalno ili na manjem broju čvorova

Kompromisi (5)

- Projektovanje zaobilaznih puteva
  - projektom sistema može da se predvidi da se u zavisnosti od potreba i trenutnih uslova žrtvuju različite stvari
    - u zavisnosti od vrste transakcija bira se šta se žrtvuje
  - npr. u slučaju prodavnice
    - nije veliki problem ako cena proizvoda *privremeno* nije ujednačena
    - osim ako se radi o veoma skupom proizvodu
    - ili je učestalost odstupanja veoma visoka
    - nije problem ako evidentirano stanje u magacinu privremeno ne odgovara stvarnom stanju
    - ako je kupac unapred obavešten o takvoj mogućnosti
    - ako se to ne dešava često i ne traje dugo
    - ali može da bude problem ako se naplati proizvod koji nije na stanju
    - ako nije moguće obezbediti novu količinu u prihvatljivom vremenskom roku
- U suštini se svodi na BASE

Zaključak

- Činjenica da nije moguće napraviti savršen distribuiran sistem (u odnosu na skup uslova ACID) ne znači da nije moguće napraviti sistem koji zadovoljava realno prihvatljive zahteve (skup uslova BASE ili drugi kompromisi)
- Mnogi veliki poslovni sistemi danas koriste distribuirane baze podataka koje počivaju na skupu uslova BASE, što potvrđuje da su relaksirani uslovi često prihvatljivi
  - ako se vodi računa pri projektovanju i
  - preciznije pristupi definisanju zahteva

Projektovanje sa BASE uslovima

- U osnovi se svodi na dve nove aktivnosti:
  - Funkcionalna dekompozicija podataka u fragmente
    - ako pretpostavljamo da se već projektuje distribuirana baza podataka, ovde samo malo preciznije određujemo uslove za definisanje fragmenata
  - Implementacija transakcija prema BASE uslovima
    - operacije se ne menjaju, ali se menja vreme njihovog izvršavanja
    - a time i način implementiranja
  - Određivanje uslova replikacije
- Postoje i drugačiji pristupi
  - ovaj je među jednostavnijim
  - i verovatno među najzastupljenijim

Projektovanje sa BASE uslovima (2)

- Dekompozicija
  - skup podataka se deli na fragmente
    - koji predstavljaju najmanje moguće funkcionalne celine
    - unutar kojih moraju da važe ACID uslovi
    - među fragmentima je dovoljno da važe BASE uslovi

Projektovanje sa BASE uslovima (3)

- Podela transakcija na sinhroni i asinhroni deo
  - svaka transakcija se *locira* u jednom matičnom fragmentu
  - promene podataka u matičnom fragmentu se implementiraju sinhrono, tj. na uobičajen način u okviru transakcije
  - promene podataka u drugim fragmentima se u okviru transakcije samo *evidentiraju*
    - evidencija mora da zadovoljava ACID uslove
  - evidentirane potrebne izmene se u drugim fragmentima sprovode asinhrono

Projektovanje sa BASE uslovima (4)

- Određivanje uslova replikacije
  - pretpostavlja se da među replikama važe BASE uslovi
  - sinhronizacija replika se odvija asinhrono, van transakcija

Projektovanje sa uslovima BASE - primer

- Fragment A: podaci o proizvodima
- Fragment B: podaci o prodaji
- Transakcija ažuriranja cene proizvoda u režimu ACID:

```
BEGIN TRANSACTION
UPDATE Product SET Price = :new_price
WHERE ProductID = :product_id
UPDATE CartItem SET Price = :new_price
WHERE ProductID = :product_id
AND CartID in (SELECT CartID FROM Cart WHERE Status = 'OPEN')
END TRANSACTION
```

- Ista transakcija u režimu BASE, locirana u fragmentu A:

```
BEGIN TRANSACTION
UPDATE Product SET Price = :new_price
WHERE ProductID = :product_id
QUEUE ADD
UPDATE CartItem SET Price = :new_price
WHERE ProductID = :product_id
AND CartID in (SELECT CartID FROM Cart WHERE Status = 'OPEN')
END TRANSACTION
```

Posledice opisanog postupka

- Svi upiti u okviru jednog fragmenta su *lokalno konzistentni*
  - u okviru fragmenata su ispoštovana sva pravila integriteta
- Rezultati upita na različitim replikama su *potencijalno* nekonzistentni
  - više replika može da vrati različite rezultate
- Rezultati logički povezanih upita na različitim fragmentima mogu da budu nekonzistentni
- Fragment je u osnovi raspoloživ i u slučaju particionisanja
  - funkcionalan je i može da odgovara na neke upite
- Svaka promena podataka se u nekom trenutku propagira i na druge replike i na druge povezane fragmente
- Baza podataka ispunjava BASE uslove

Konflikti

- Predstavljaju osnovni vid problema
  - kod svih vidova implementacije BASE skupa uslova
- Različite vrste konflikata
- Osnovni oblik
  - ako se dve replike istog podatka nezavisno menjaju, pitanje je koja verzija je "*ispravna*" i kako uspešno izvesti usklađivanje
- Složeniji oblik
  - ako se koriste i redundantni podaci a ne samo replike

Razrešavanje konflikata

- Više načina rešavanja:
  - zabrana menjanja podataka u slučaju particionisanja
    - umanjena raspoloživost
  - definisanje hijerarhija među redundantnim kopijama konkretnih vrsta podataka
    - definisanje primarnih i sekundarnih fragmenata za sve podatke
    - samo na primarnim se izvode transakcije u odnosu na te podatke
    - na sekundarnim se samo propagiraju (odložene) izmene
    - u primeru sa cenama, za cenu je primarni fragment A
    - ne umanjuje upotrebljivost, ali komplikuje implementaciju
  - višestruke verzije podataka (MVCC – multiversion concurrency control)
    - alternativa zaključavanju
    - za svaku kopiju podatka se vodi verzija
    - može da postoji više verzija "istog" podatka
    - povezano sa konceptom ***optimističkih transakcija***
    - konflikti mogu automatski da se prepoznaju, ali ne i da se otklone
  - ...

Literatura za ovu temu

- Ozsu, Valduries – Principles of Distributed Database Systems (2.ed) (1999)