3 - Klijent server tehnologija

S A D R Ž A J

3.1 Nastanak klijent server tehnologije

- Model klijent server sistema predstavlja nastavak razvoja otvorenih sistema koji su dominantna organizacija računar.sistema i aplikacija.
- Zahvaljujući novim tehnologijama i sve većem razvoju PC računara, vršena je selidba aplikacija sa *Mainframe* računara na desktop računare
- Četiri uslova koja su doprinela velikom razvoju:
  1. veliki pad cena procesora i računarskih komponenti,
  2. ogromno povečanje resursa računarskih komponenti,
  3. veliki broj prilagođenih softverskih standarda,
  4. veliki razvoj objektno orjentisanih tehnologija.
- Organizacije danas očekuju da za svoja ulaganja dobiju sve veće vrednosti i sve više podataka koji će biti pravovremeni i tačni.
- **Brzina dobijanja podataka** je danas odlučujući faktor
- Veliki broj različitih tehnologija kao što su: objektno orjentisani razvoj, relacione i objektno orjenitisane baze podataka, multimedija, ekspertni sistemi, geografski informacioni sistemi (GIS), tekstualni procesori, prepoznavanje glasa i td.

3.1 Nastanak klijent server tehnologije

- Klijent server model organizacije omogućavao je idealnu platformu koja je objedinjavala ove mogućnosti u samo **jednoj običnoj mašini.**
- Zahvaljujući razvoju svih ovih tehnologija došlo je do kompletnog zaokreta tj. reinženjeringa poslovnog procesa.

1. ***Globalizacija*** - svet polako postaje jedna velika pijaca jer se mnoge barijere brišu. Kvalitet, cena, različiti proizvodi i servisi postaju ključni faktori i najvažniji marketinški prioriteti.
2. ***Decentralizacija upravljanja*** - kvalitet i fleksibilnost zahtevali su donošenje odluka od strane mnogih individualaca koji su u direktnom kontaktu sa kupcima. Nove tehnologije su morale da omoguće i takvim osobama da donesu pravilnu odluku o kojoj će poslovodsvo biti brzo informisano. **Brzina odlučivanja** postaje presudni faktor za napredak.
3. ***Mrežno upravljanje*** - kako se biznis vodio sa velikog broja decentralizovanih lokacija, tehnologija je morala da omogući da se sve te decentralizovane lokacije povežu u jednu i da predstavljaju jednu jedinstvenu centralizovanu jedinicu.

3.1 Nastanak klijent server tehnologije

4. ***Dostupnost informacija*** - da bi neki proizvod imao dobru prođu mora se obezbediti da informacije o njemu budu dostupne svima.
5. ***Niska cena*** - zahvaljujući standardizaciji proizvoda omogučeno je velikom broju firmi da proizvode isti proizvod tako da je velika ponuda smanjila cenu proizvoda a povećala kvalitet.
6. ***Različite hardverske platforme*** - zbog raznolikosti *hardware* trebalo je omogučiti da se svi oni međusobno vide, tj. da mogu da komuniciraju
7. ***Minimalno poznavanje tehnologije*** - trebalo je omogučiti jednostavan pristup svim informacijama, kao i da se vreme obuke za rad svede na najmanju moguću meru -tehnologije vizuelnog pristupa informacijama
8. ***Različita softverska rešenja*** - ogroman razvoj softvera koji je morao da nahrani računare sa odgovarajučim programima zahtevao je i različite softverske alate koji su opet morali da budu kompatibilni
9. ***Različite baze podataka***-velika količina podataka uslovila je razvijanje i odgovarajučih softverskih alata za unošenje, ažuriranje i pregledanje. SQL (*Structured Query Language*) predstavlja jedan od osnovnih jezika za rad sa bazama koji podržavaju gotovo sve baze podataka

3.2 Pojam klijent server

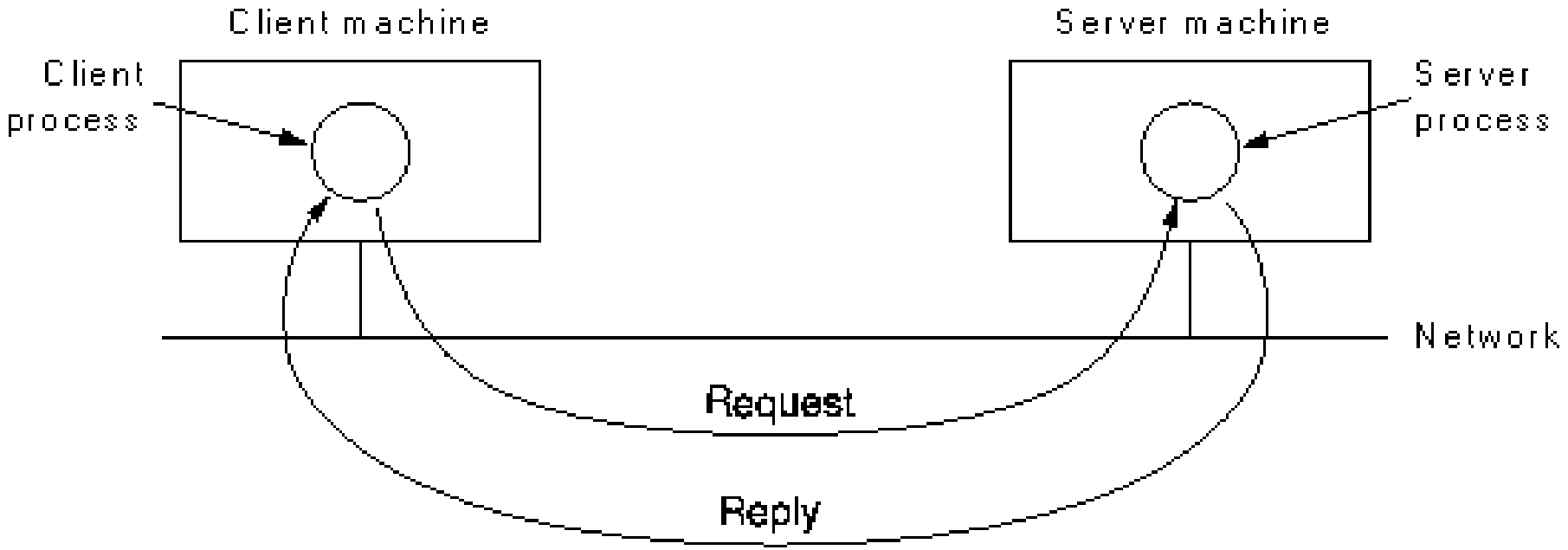
- Klijent-server arhitektura i klijent-server model obrade podataka javlja se krajem **XX veka** (1990) a do pravog razvoja dolazi u **XXI veku**
- Razvijen je kao rezultat nastojanja da se što bolje iskoriste resursi PC-a i njihovom integracijom u jedinstven sistem za obradu podataka.
- Klijent-server predstavlja arhitekturu gde su korisnik- **klijent** i davalac usluga - **server** odvojeni ili neravnopravni (Primer : WEB servisi)
- Klijent je **aktivan** korisnik, koji šalje zahteve i čeka dok se isti ne ispune, dok je server **pasivan**, čeka na zahteve, izvršava ih i odgovara.
- Serveri su obično veoma jaki računari s dobrim karakteristikama zbog toga što istovremeno moraju obraditi mnogo zahteva koji stalno rastu
- Servere pokreću i posebni mrežni OS, za razliku od običnih klijent OS
- Klijent/server model je baziran na distribuciji funkcija između dva tipa nezavisnih i autonomnih procesa: servera i klijenta.
- Klijent je bilo koji proces koji zahteva specifične usluge od server procesa dok je server proces koji osigurava usluge za klijenta.
- Klijent i server mogu biti smešteni u **istom ili u različitim** računarima

3.2 Pojam klijent server

- U slučaju da su klijent i server procesi smešteni u dva ili više nezavisnih i umreženih računara, server proces može osigurati usluge za više od jednog klijenta.
- Pored toga, klijent može zahtevati usluge i od više servera iz okruženja bez obzira na **njihove lokacije** ili **fizičke karakteristike** računara
- Klijent-server model obrade podataka je model distribuirane obrade podataka, gde se funkcije jednog korisničkog programa raspodeljuju na najmanje dva procesa koji međusobno komuniciraju.
- Ta dva procesa obrade podataka su: klijentski i serverski.
- Oba procesa se mogu izvršavati na jednom računaru, ali pun opseg mogućnosti se realizira distribucijom ovih procesa na veliki broj, **fizički dislociranih računara**.
- Elementi koji čine klijent-server sistem su: jedan ili više klijenata i servera, klijentski i serverski procesi kao i komunikacijski međusloj

Za punu primenu klijent-server arhitekture, potrebna su najmanje dva računara, povezana komunikacijskom mrežom, sa instaliranim mrežnim OS i uspostavljenom komunikacijom.

3.3 Način funkcionisanja



1. Server proces započinje na nekom računaru (na kojem je smešten), pokreće se, a zatim prelazi u ***sleep*** mod i čeka da ga neki klijent proces kontaktira i zatraži neku uslugu od njega.
2. Klijent proces započinje na istom ili nekom drugom računaru koji je preko mreže povezan sa serverom. Procesi se često pokreću od strane interaktivnih korisnika koji zahtevaju izvršenje određenih naredbi.
3. Kada server proces završi posao (servis) koji je od njega zatražen od strane klijenta, prelazi ponovo u *sleep* mod i čeka sledeći zahtev

3.3 Način funkcionisanja

- Klijent predstavlja jednokorisničku radnu stanicu koja omogučava predstavljenje traženih informacija u vidu izračunavanja, povezivanja ili servisa baze podataka koji se odnose na poslovne zahteve.
- Server predstavlja radnu stanicu sa jednim ili više procesora koji omogučava višekorisnički (multiuser) rad, koji treba da omogući što **brži odziv** i **davanje tražene informacije** klijentima.
- Ideja ovog modela je da se napravi kooperacija procesa između servera koji obezbeđuju servise i klijenata koji servise zahtevaju.
- Ova paradigma se koristi na nivou aplikacije kao i na nivou OS
- Svaki proces u iterakciji igra ili ulogu servera ili ulogu klijenta.
- Proces koji se ponaša kao server može zahtevati servis od drugog servera da bi izvršio neki zadatak, time dobijajući i ulogu klijenta.
- Mašina može biti jedan klijent, jedan server, više klijenata ili servera ili mešavina klijenata i servera.

3.3 Način funkcionisanja

- Logička komunikacija između klijenata i servera je bazirana na razmeni zahteva i odgovora i to na sledeći način:
- Fizički, zahtevi i odgovori se prosleđuju dotičnom kernelu, i šalju kroz komunikacionu mrežu ciljnim kernelima i procesima.
- Iz jednostavnosti ovog modela proizilazi potreba za samo dva komunikaciona sistemska poziva kernelu:

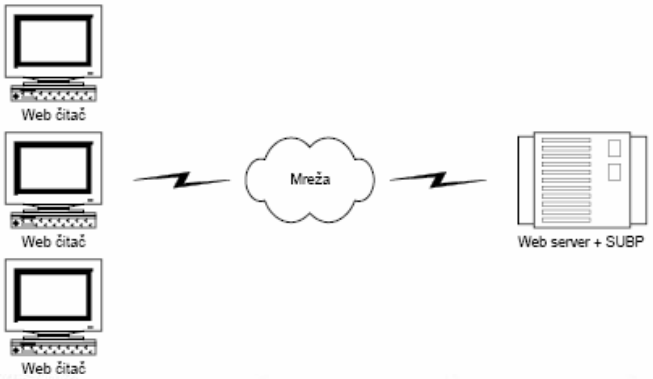
  send (pošalji) i **receive** (primi)
- Važno pitanje je kako klijent locira server ako on promeni lokaciju ili postoji više servera u mreži.
- Zbog fleksibilnosti serveri se identifikuju preko imena.
- Vezujući serveri (***binder servers***) spajaju klijente i servere.
- Procedura koja izvršava ovo spajanje nazva se dinamičko povezivanje (***dinamic bindig***).
- Kada se server startuje on šalje povezivaču podatke o svojoj lokaciji i na osnovu tih podataka klijenti znaju lokaciju odgovarajućeg servera.
- Povezivači mogu pomagati u autentifikaciji klijenata za servere.

3.4 Komponente klijent server modela

- Klijent/server arhitektura se zasniva na hardverskim i softverskim komponentama koje su međusobno povezane i na taj način čine sistem.
- Svaki klijent server sistem sadrži tri komponente:

1. **Klijent** - predstavlja bilo koji računarski proces koji zahteva usluge od servera. Klijent, poznat još i kao glavna (***front-end***) aplikacija, odražava činjenicu da je krajnji korisnik obično u interakciji sa klijent procesom sa kojim započinje neku aktivnost.
2. **Server** – to je bilo koji računarski proces koji čeka na zahteve od klijenata i osigurava potrebne usluge za klijente shodno pristiglim zahtevima. Poznat je i kao pozadinska (***back-end***) aplikacija.
3. **Komunikacioni posrednik** - označava bilo koji računarski proces čijim posredstvom klijent i server komponenta mogu da međusobno komuniciraju.

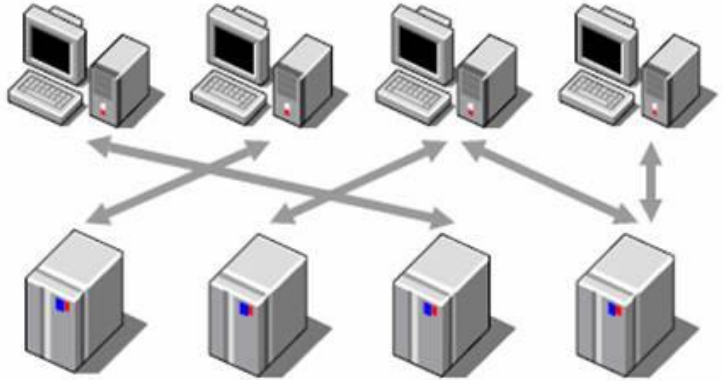
3.4 Arhitektura klijent server modela



DVA SLOJA

Korisnički interfejs
+deo za upravljanje procesima

Upravljanje podacima (DBM)
+deo za upravljanje procesima

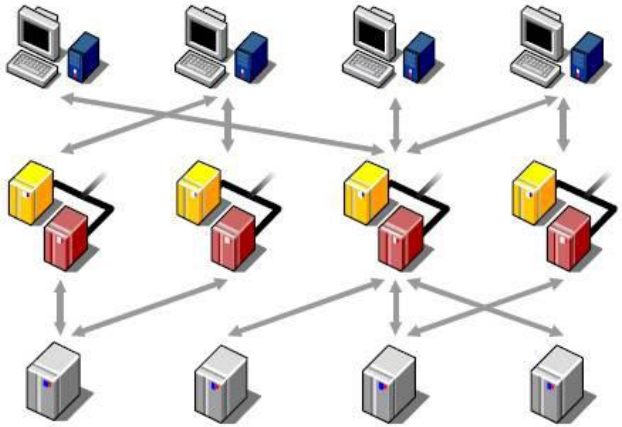


TRI SLOJA

korisnički interface

upravljanje procesima

upravljanje podacima
(DBM)



3.4 Primer: traženje podataka iz baze podataka

Primer: *Za ilustraciju interakcije komponenata može da posluži primer kada klijent zahteva servise od procesa baze podataka. Izvršenje aplikacije je razdvojeno na dve glavne i nezavisne komponente, klijent i server, a komunikacioni posrednik omogućava klijent i server procesima da rade zajedno.*

1. Najpre klijent proces šalje SQL zahtev do komunikacionog posrednika.
2. Komunikacioni posrednik prosleđuje SQL zahtev do server procesa za baze podataka koji prima zahtev.
3. Obrada SQL zahteva se vrši na serveru. Server proces potvrđuje izvršavanje zahteva i počinje sa njegovim izvršavanjem.
4. Potom server šalje selektovane podatke komunikacionom posredniku koji ih prosleđuje i formatira za prenos i šalje ih klijentu.
5. Komunikacioni posrednik obezbeđuje da poruke između klijenta i servera ispravno putuju kroz mrežu i stignu na svoje odredište.
6. Klijent proces je odgovoran za interfejs krajnjeg korisnika, neku proveru lokalnih podataka,njihovu pripremnu obradu i prikaz podataka

3.5 Osnovni zahtevi

- **Hardverska nezavisnost** je postignuta kada klijentski, serverski i procesi komunikacionog međusloja uspešno funkcionišu na različitim hardverskim platformama bez ikakve funkcionalne razlike.
- **Softverska nezavisnost** je postignuta kada klijentski, serverski i procesi kom.posrednika mogu funkcionisati na različitim OS uz upotrebu različitih mrežnih protokola kao i da podržavaju različite aplikacije.
- **Otvoren pristup servisima** je ostvaren kada klijentski proces ima mogućnost da, po potrebi, bez ograničenja, koristi usluge svih serverskih procesa u mreži, nezavisno od lokacijske udaljenosti klijenta i servera.
- **Interoperabilnost i integracija** procesi klijenta i servera moraju da budu integrisani u "bezšavnu" formu sistema, tj. da različite aplikacije imaju mogućnost razmene podataka između različitih OS i hardverskih platformi bez obzira na udaljenost, opremu i tip OS.
- **Standardizacija** da osigura što veći stepen hardverske/softverske nezavisnosti i nezavisnost korisničkog programa od podataka. Korisnički *interface*, pristup podacima i mrežni protokol, kao i drugi elementi klijent-server arhitekture trebaju biti pokriveni standardima.

3.5 Osnovni zahtevi

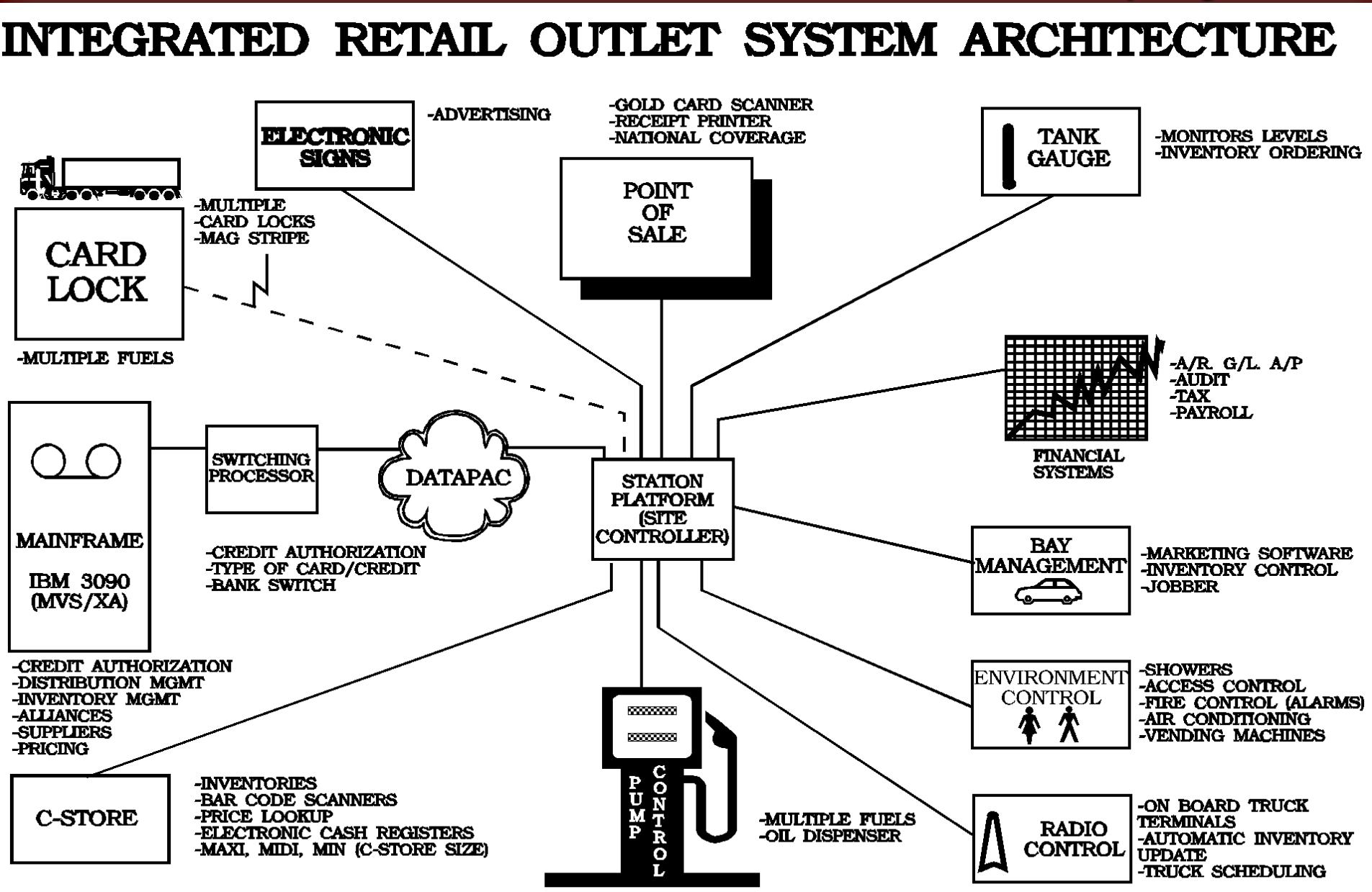
- **Funkcionalna distributivnost** je ostvarena kada klijentski, serverski i procesi međusloja predstavljaju nezavisne činioce u obradi podataka i imaju precizno definisane namene, zadatke i granice. Funkcionalna distributivnost treba da omogući "**zamenjivost**" delova softvera, što podrazumeva da tekuća verzija nekog klijentskog, serverskog ili programa u međusloju klijent-server arhitekture, može zameniti novom, a da to ne utiče na funkcionisanje ostalih komponenti softvera. Celokupna obrada podataka je distribuirana između klijenta i servera.
- **Podela opterećenja obrade** mora se povinovati sledećim zahtevima:

1. jasno definisanje funkcionalnosti obe strane - klijent i server procesi moraju biti autonomni, sa jasno definisanim granicama i funkcijama.
2. lokalno korišćenje resursa (klijenta i servera) je maksimalno tj. dodela zadataka računaru koji najviše funkcionalno odgovara tom zadatku.
3. laka prenosivost procesa - da mogu biti lako i što bolje izvršeni na više snažnih različitih hardverskih platformi.
4. prenosivosti softvera između različitih mašina bez potrebe da se interveniše na izvornom kodu, već samo prevođenje i povezivanje

3.6 Prednosti klijent server tehnologije

- Osnovna prednost višeslojne klijent-server arhitekture je ta što bazu podataka čini nezavisnom od aplikacija, jer integritet baze podataka osigurava njenu nezavisnost od aplikacija koje nad njom rade.
- Jedini mehanizam potreban za komunikaciju aplikacije sa bazom podataka je SQL upit, koji se prosleđuje direktno ili kroz neku mrežu.
- Nedostatak ove arhitekture što se veći deo aplikacije veže za klijenta.
- Povezivanje dinamičkog ponašanja sistema za klijenta, zbog čestih izmena u korisničkom okruženju, prouzrokuje skupo praćenje sistema a zbog čestih izmena u tehnologiji korisn. interfejsa skupo održavanje
- **Osnovne prednosti** koje nam klijent server arhitektura omogućava:
  1. **Pristup ogromnoj količini podataka**
  2. **Integrisani servisi na jednom mestu**
  3. **Deljeni resursi na različitim platformama**
  4. **Zamenljivost podataka i interoperatibilnost**
  5. **Maskirani fizički pristup podacima**
  6. **Nezavisne lokacije procesiranja i pamčenja podataka**
  7. **Centralizovano upravljanje**

2.4 Računarske mreže - osnovni pojmovi



Hvala na pažnji !!!

Pitanja

? ? ?