

ВИСОКА ШКОЛА

ЕЛЕКТРОТЕХНИКЕ И РАЧУНАРСТВА

СТРУКОВНИХ СТУДИЈА

11000 Београд, Војводе Степе 283.

Klijent Server Sistemi

Skripta sa 166 pitanja

djwibe, 09.11.2007

ISPITNA PITANJA IZ PREDMETA KLIJENT-SERVER SISTEMI: od Juna 2007	rok	rok
---	-----	-----

1. Osnovna terminologija (IT, IS, DBMS, BP), Oblasti primene IT 2. Kategorije problema podesne za rešavanje uz pomoć IT, Oblici primene IT 3. Evolucija odnosa aplikacije, OS i H/W 4. Generacije programskih jezika 5. Generacije H/W 6. Etape u razvoju sistema za upravljanje podacima 7. Obrada zasnovana na fajlovima 8. Obrada zasnovana na korišćenju BP 9. ANSI-SPARC arhitektura 10. Kategorije savremenih DBMS 11. Relacione BP 12. Multimedijalne BP 13. Sistemi za upravljanje bazama podataka (DBMS) 14. DBMS-osnovne funkcije 15. DBMS-osnovne komponente 16. DML funkcije 17. DDL kompajler 18. Funkcije SQL DB servera 19. Arhitektura SQL DB servera 20. Izbor arhitekture SQL DB servera 21. Arhitektura RDBMS Oracle10g 22. Detaljna arhitektura Oracle10g servera 23. Oracle10g procesi 24. Oracle instanca 25. Logička struktura Oracle BP 26. Fizička struktura Oracle BP 27. Arhitektura RDBMS My SQL 28. Objektno-relacioni DBMS 29. Objektno-orijentisani DBMS 30. Modeli podataka 31. SQL i programski interfejsi-Evolucija 32. Programski interfejsi BP 33. Ugnježdeni API (ESQL API), funkcije, obrada 34. Obrada ugnježdenog SQL API-ja 35. DB interfejsi na nivou poziva (CLI API) 36. Komparacija ESQL API i CLI API 37. Osnovi upravljanja procesima 38. Upravljanje transakcijama 39. Svojstvo ACID 40. Kratke i duge transakcije, definicija i komparacija 41. Upravljanje konkurentnim pristupom 42. Problemi konkurentnosti (transakcija) 43. Pesimistički pristup upravljanju konkurentnim radom 44. Nivoi zaključavanja objekata BP 45. Upravljanje transakcijama primenom TPM		
	Oct-07	

46. Pojava i razrešavanje dead-lock-ova	
47. Upravljanje sigurnošću baze podataka	
48. Problemi sa transparentnošću procesa	
49. Distribuirani sistemi (definicije, prednosti, nedostaci)	
50. Distribuirana obrada, distribuirana baza i distribuirani DBMS	
51. Tehnike distribucije podataka	
52. Horizontalno i vertikalno particioniranje podataka iz BP	
53. Keširanje podataka	Oct-07
54. Ekstrakt i replika	
55. Replika kod MySQL-a	Oct-07
56. Korišćenje replikacionih servera	
57. Načini distribucije procesa	
58. Udaljeni zahtev	
59. Udaljena transakcija	
60. Distribuirani zahtev	
61. Distribuirana jedinica posla	
62. Protokol dvofaznog izvršavanja	
63. Osnovna svojstva distribuiranog DBMS	
64. Multidatabase sistemi	
65. Federalizovani MDBMS sistemi	
66. Arhitektura federalizovanih MDBMS sistema	
67. Vlasništvo nad podacima	
68. Modeli obrade podataka	
69. Arhitekture informacionih sistema	
70. Centralizovana višekorisnička arhitektura	
71. Distribuirana jednokorisnička arhitektura	
72. Distribuirana višekorisnička arhitektura	
73. Tipovi (kategorije) IS, aplikacija i korisnika	
74. Sistemi OLTP tipa	
75. Sistemi DSS tipa	
76. Prednosti i nedostaci distribuiranih IS	
77. Arhitekture distribuiranih IS (DIS)	
78. Master-slave model	
79. Klijent-server model	
80. Model od-sloja-do-sloja (P-to-P)	
81. Grupni model	
82. Model distribuiranih objekata	
83. Distribuirani objekti i komponente	
84. Java Beans komponentni model	
85. Objektni transakcioni monitor (OTM)	
86. Komponente serverske strane	
87. Enterprise Java Beans-i	Sep-07
88. Poslovni objekti	
89. 3-slojna c/s arhitektura sa distribuiranim objektima	
90. EJB i CORBA	
91. Java i Java arhitektura	
92. Java platforme (J2EE)	
93. Model multimedijalnog toka	

94. Klasifikacija klijent-server sistema prema hijerarhiji, lokaciji	
95. Klasifikacija klijent-server sistema prema nameni servera	
96. Fajl server	
97. DB server	
98. Transakcioni server	
99. Kanalisanje i TPM	
100. Standardi za TP monitore	
101. Komparacija TPM i memorijskih procedura	
102. Objektni aplikacioni server	
103. Grupni server	
104. Web aplikacioni server	
105. Komponente distribuiranih IS	
106. Dvoslojna C/S arhitektura	
107. Troslojna C/S arhitektura	
108. Komparacija 2-slojne i 3-slojne arhitekture	
109. Funkcionalna dekompozicija monolitnih aplikacija	
110. Moguća alokacija funkcija na elemente C/S sistema	
111. Osnovne C/S tehnologije	
112. Tehnološki elementi C/S sistema	
113. Generalizacija s/w arhitekture klijent-server sistema	
114. Tehnologije servera (arhitektura s/w servera, h/w platforma)	
115. OS servera	
116. Bazni servisi OS	
117. Prošireni servisi OS	
118. Tehnologije klijenta (arhitektura sw klijenta, h/w platforma)	
119. OS klijenta	
120. Korisnički interfejs klijenta	
121. Alati za razvoj C/S aplikacija	Oct-07
122. Struktura i elementi savremenog razvojnog alata	
123. Izbor alata za razvoj C/S aplikacija	
124. "Bilderi" vizualnih mesta za 1-sloj	
125. "Bilderi" poslovnih objekata za 2-sloj	
126. Okviri (frameworks) za 3-ći sloj	
127. Integrisana razvojna okruženja (IDE)	
128. Primeri proizvođačkih IDE-ova	
129. Primeri otvorenih IDE-ova	
130. MS .NET okvir (Framework)	
131. Srednji sloj (middleware) C/S sistema, namena, položaj, servisi	
132. Celovite arhitekture srednjeg sloja, sličnosti i razlike	
133. OSF DCE	
134. OSF DCE arhitektura	
135. Distribuirani Naming servisi	
136. Tipovi middleware-a	
137. Srednji sloj RPC tipa	
138. MW RPC-prezentacioni sloj	
139. MW RPC-sloj sesije	
140. Srednji sloj MOM tipa	
141. Transakciono orijentisan MW (transakcioni monitori)	

142. Transakciono orijentisan MW (aplikacioni serveri)		
143. MW baziran na distribuiranim objektima		
144. Slojevi MW baziranog na distribuiranim objektima		
145. OMG OMA (Objct Management Architecture)		
146. Middleware tipa OMG CORBA		
147. Centralni deo CORBA arhitekture		
148. Povezivanje/integracija aplikacija u preduzeću (EAI)		
149. EAI na nivou procesa		
150. Brokeri poruka		
151. Srednji sloj za povezivanje sa BP (namena, položaj i funkcije)		
152. Načini povezivanja klijenta i servera		
153. Native/proizvodjački DB MW		
154. Arhitektura c/s sistema sa nativnim DB MW u homogenom okruženju		
155. Arhitektura c/s sistema sa nativnim DB MW u heterogenom okruženju		
156. Implementacije tipa zajedničkog interfejsa		
157. Povezivanje sa ODBC		
158. Povezivanje sa JDBC		
159. JDBC, osnovni delovi	Sep-07	Oct-07
160. JDBC tip 1 i 2, arhitektura		
161. JDBC tip 3 i 4, arhitektura		
162. Povezivanje sa OLE-DB i ADO		
163. MS ADO.NET		
164. Arhitektura ADO.NET		
165. Komponente MS ADO.NET aplikacije		
166. Položaj i odnos ADO i ADO.NET		
167. Povezivanje putem zajedničkog gateway-a	Oct-07	
168. Povezivanje putem zajedničkog protokola		
169. Otvoreni sistemi i IT standardi, standardizacije organizacije		

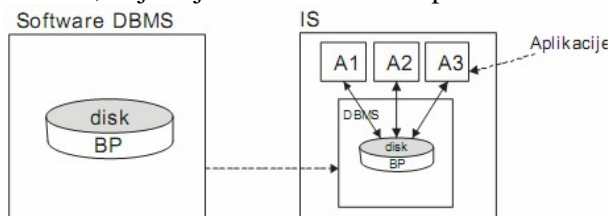
	Pitanja koja su u skripti
	Pitanja kojih nema u skripti

U kolonama “ROK” se nalaze rokovi u kojima su doticna pitanja bila na ispitu.
Na kraju skripte se nalazi i spisak “starih” 144 pitanja i rokovi u kojima su izlazili poslednjih godinu dana.

Srecno, u spremanju ispita,
djwibe

1. Osnovna terminologija (IT, IS, DBMS, BP), Oblasti primene IT

- IT- skup metoda i sredstava iz oblasti racunarstva i TK(ICT) koji pomažu u prikupljanju, cuvanju, obradi i prenosu informacija. ICT - Information Communication Technologies, TK - Telecommunications
- IS- Informacioni sistem je sistem u kome se veze između objekata i veze sistema sa okolinom ostvaruje razmenom informacija.
- DBMS – je s/w sistema za cuvanje i korišćenje podataka, uz logicku i fizicku nezavisnost programa od podataka, i jednostavan jezik pristupa BP. DBMS (SUBP) - Data Base Management System s/w – Software
- BP (DB-Database) – Baza Podataka je skup međusobno povezanih podataka, sa minimumom redundanse, koje zajednicki koriste svi procesi obrade (Aplikacije) u sistemu.



Oblasti primene IT: finisijske institucije, trgovina, transport, obrazovanje, nauka, vojne namene, industrija, energetika, drzavna administracija, uprava, medicina.

2. Kategorije problema podesne za rešavanje uz pomoć IT, Oblici primene IT

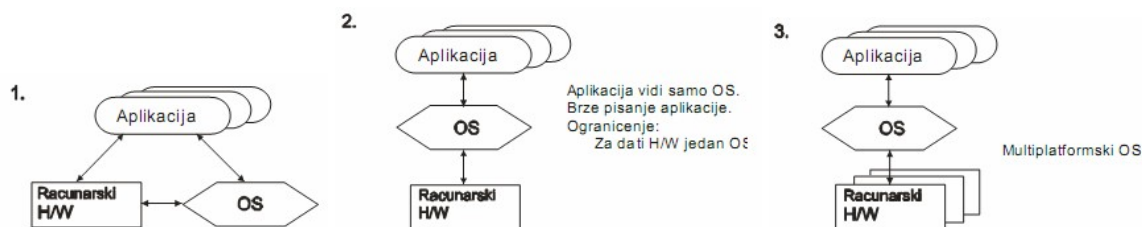
Kategorije problema podesne za rešavanje na racunaru:

- Finalni rezultat mora biti ekonomski opravdan
- Jasno definisan i sa dostiznim ciljevima
- Repetitivan
- Kolicina podataka znacajna (posl. apl.)
- Brojni / Slozeni proracuni (Naucno-Teh. apl.)

Oblici primene –Brojne moguće klasifikacije

- Ugradeni s/w (embedeed s/w) u uredaje
- Merni sistemi
- CAD / CAN / CIM (Computer Integrated manufacturing)
- Informaciono-Upravljacki Sistemi
- Simulacioni Sistemi
- Simulaciono-Trenažni Sistemi
- Informacioni Sistemi, razni: Poslovni: *Management IS*, Tehnicki: *GIS - Geographic IS*, *EIS - Enterprise (za kompaniju) IS*, *Executive (za direktore) IS*, *DSS - Decision Support System*, *IIS - Intelligent IS*

3. Evolucija odnosa aplikacije, OS i H/W



4. Generacije programskih jezika

1. generacija(1GL): 45.-60.

Mašinski+Simbolicki Jezik (Asembler)

2. generacija(2GL): 55.-75.

Visokog nivoa, Proceduralni, (Fortran 55/77/90, Cobol 59, Algol 60, Basic 64)

3. generacija(3GL): od 65. na dalje.

Jezici opšte namene C 70. Pascal 71., Ada 80./95.,...) specijalni (Lisp, Prolog), OOJ (C++ 83., Java 95.- Platform Independent)...

4. generacija(4GL): od 75.

Proceduralni (*Specificira se nacin na koji dolazimo do rešenja.*) i Neproceduralni (*Zadaje se samo željeni rezultat.*), Upitni (SQL), generatori koda (CASE), DS jezici

5. Generacije H/W

Moor-ov Zakon: Dupliranje performansi μP svakih 18 meseci, a broja komponenata na 2 godine.

I.generacija: 1942.-1950.

- Elektronske (vakuumске) cevi
- Feritne memorije
- Magnetni doboš
- Mašinski jezik
- ENIAC 1945., UNIVAC I 1951. (30 tona), IBM 650 1953. (masovna proizvodnja).

II.generacija: 1959.-1965.

- Poluprovodnicki elementi
- Magnetne trake
- Mašinski + Simbolicki Jezik (Asembler)
- PRIMERI: IBM 1130, IMP Cer.

III.generacija: 1965.-1970.

- Integrisana Kola
- Magnetne diskovi i trake
- Viši Programski Jezici
- Time-Sharing, multiprogramming.
- PRIMERI: IBM System/360

IV.generacija: 1970.-1980.

- Monolitna Integrisana Kola (veci stepen integracije)
- Virtualna memorija
- Rast Brzine
- PRIMERI: IBM System/360

V.generacija: 1982.-1990.

- Na bazi koncepta Veštacke Inteligencije (Artificial Intelligence)
- Dalji rast stepena Integracije
- Brži, Performantniji, Pouzdaniji, sa Manjim Dimenzijama

6. Etape u razvoju sistema za upravljanje podacima

- Programsko upravljanje zapisima
 - Programsko / sekvencijalno upravljanje zapisima (1955-1970),
 - Tradicionalna obrada zasnovana na fajlovima,
 - Obrada zasnovana na korišćenju BP
- On-Line BP (Hijerarhijski/ Mrežni model)
 - Sistem je ON-LINE da bi se ukazala razlika i pomak u odnosu na I.etapu. U prvoj etapi nismo imali ažurno stanje sem na nivou fajla. I generacija DBMS

- Hijerarhijski DB model: Definiše hijerarhijski uređene podatke, Prikaz u obliku stabla 1:M, Veze pomoću pointera (fizička adresa rekorda, veze između rekorda).
- Mrežni model: Definiše podatke organizovane u mrežu, Relacije tipa M:N, eksplicitno – pointer (N roditelja – M potomaka), Navigacija: set oriented record model
- Nedostaci: Složeno održavanje, redundansa, složen i dug razvoj BP.
- **Relacione BP**
 - Entiteti i relacije prikazani na uniforman način, putem TABELA.
 - BP – TABELA – REKORDI (redovi) – ATRIBUTI (kolone)
 - Svaka kolona ima tip podatka, ograničen broj (tipova podataka)
- **Multimedijalne BP**
 - Multimedijalni podaci: Tekst, Grafika, Slika, Audio, Video, njihova kombinacija, Bogati informacijama, Dimenzije!
 - Multimedijalne BP su III. generacija DBMS-ova.
 - Objektno-Relacioni sistemi za upravljanje BP (OR DBMS, ER DBMS (Extended))
 - Objektno-Orjentisani sistemi za upr. BP (OO DBMS).

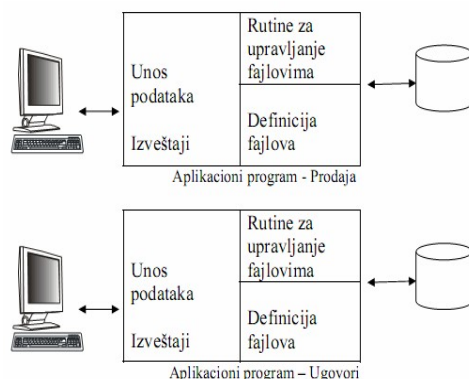
7. Obrada zasnovana na fajlovima

Fajlovski bazirani sistem predstavlja skup aplikacija koje obavljaju neke servise za krajnjeg korisnika. Ograničenja tradicionalnih – Fajlovski organizovanih sistema su:

- Razdvojenost i izolovanost podataka (Svaka aplikacija ima svoj ulaz i izlaz)
- Dupliciranje podataka (Problem održavanja višestrukih podataka.)
- Zavisnost podataka od fizicke strukture
- Nekompatibilnost fajlova
- Fiksni upiti
- Veliki porast broja apl. programa zbog velikog broja izveštaja.

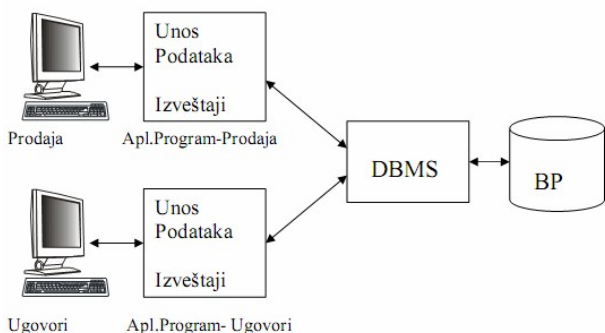
Ograničenja su posledica:

- Definicija podataka je ugnježdjena u apl. programu
- Ne postoji kontrola pristupom i manipulacijom podacima osim one koju nameće aplikacioni program.



8. Obrada zasnovana na korišćenju BP

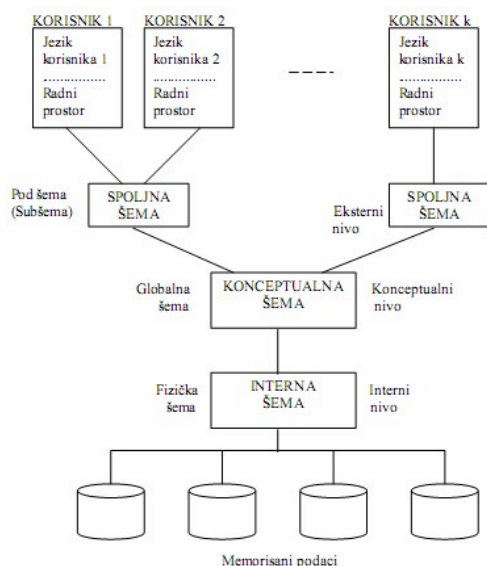
- Baza podataka je zajednicki koriscena kolekcija logicki povezanih podataka i opisa tih podataka, projektovana da zadovolji potrebe preduzeca. To je samoopisana kolekcija integrisanih rekorda (metapodaci). Ove osobine obezbedjuju BP nezavisnost programa od podataka jer je definicija podatak izolovana od aplikacije.
- DBMS je software-ski sistem koji omogucava korisnicima da definisu, kreiraju, odrzavaju i kontrolisu pristup BP. DBMS je sloj izmedju BP i aplikacije, I nema pristupa BP bez prolaska kroz DBMS.



9. ANSI-SPARC arhitektura

ANSI / SPARC arhitektura je Tro – nivojska arhitektura sa sistemskim katalogom, koji omogućava Logicku i Fizicku nezavisnost podataka, što je krucijalni koncept BP (Subšema, Logicka Šema, Fizicka Šema.).

- Eksterni nivo – korisnik vidi BP kroz svoju Subšemu. Subšema ima onoliko koliko razlicitih aplikacija radi nad bazom.
- Konceptualni nivo – Opis svih podataka u bazi (Šta sve treba da sadrži BP). Pravi je i održava DBA (Data Base Administrator).
- Interna šema – Organizuje podatke na fizickom medijumu.
- Eksterno-konceptualno preslikavanje omogućava logicku nezavisnost podataka
- Konceptualno-Interno preslikavanje omogućava fizicku nezavisnost seme.



10. Kategorije savremenih DBMS

- Relacioni sistemi za upravljanje BP – RDBMS (Entiteti i relacije prikazani na uniforman nacin, putem TABELA.)
- Objektno – Relacioni sistemi za upravljanje BP – OR DBMS (Sva perzistentna informacija je u tabelama, ali neki elementi tabela imaju bogatiju strukturu, ADT (Abstract Data Types))
- Objektno – Orjentisani sistemi za upravljanje BP – OO DBMS (Objektne BP koriste model podataka koji ima OO koncepte– klasa koja obezbeduje identifikatore objekata (OID) za svaku perzistentnu instancu klase, enkapsulaciju, višestruko nasleđivanje i podržava ADT.)
- Kategorizacija je izvršena na osnovu:
 - Modela podataka
 - Upitnog jezika
 - Racunskog modela / navigacija pristupom.

11. Relacione BP

- Entiteti i relacije prikazani na uniforman nacin, putem TABELA.
- BP – TABELA – REKORDI (redovi) – ATRIBUTI (kolone)
- Svaka kolona ima tip podatka, ogranicen broj (tipova podataka)
- Relacije implicitne – Spoljni KLJUC
- Jedinствен jezik za Definisane podataka, navigaciju i manipulaciju (SQL)
- Standardizovan Strukturirani Upitni Jezik (SQL) – Neproceduralan.
- Odgovarajući Standardi: ANSI, ISO (SQL-1: 1986/1989, SQL-2: 1992, SQL-3: 1999)
- Stored-Procedure - stoje u BP i pokreću se kroz katalog. Smanjuje se vreme pristupa, duže je kad su procedure u aplikaciji.
- Podaci se pretražuju na bazi vrednosti u Polju.
- Pregled rezultata Upita se vrši pod kontrolom kursora.
- Primeri: Oracle 7, MS SQL Server, DB2, Sybase Sys. 10/11, Informix, Ingres...

12. Multimedijalne BP

- Multimedijalni podaci:
 - Tekst, Grafika, Slika
 - Audio, Video, njihova kombinacija
 - Bogati informacijama, Dimenzije! (zauzeće memorije)
- Ekspanzija Multimedijalnih podataka (doprinese Internet)
- U BP su meta–podaci, sami podaci su u fajlovima na OS. (meta–podaci – podaci o podacima).
- Multimedijalne BP su III.generacija DBMS–ova.
 - Objektno-Relacioni sistemi za upravljanje BP (OR DBMS, ER DBMS–Extended Relational DBMS)
 - Objektno-Orjentisani sistemi za upr. BP (OO DBMS).

13. Sistemi za upravljanje bazama podataka (DBMS)

- Sistemi za upravljanje BP (SUBP/DBMS) obezbeđuju:
- Mogućnost definisanja BP putem DDL–a
- Mogućnost da korisnici ubacuju, ažuriraju, brišu i pretražuju podatke u BP putem DML–a.
- Kontrolisani pristup podacima
 - Katalog dostupan korisniku
 - Sigurnost
 - Integritet
 - Kontrola konkurentnosti, oporavka,...

14. DBMS-osnovne funkcije

Funkcije DBMS–a su

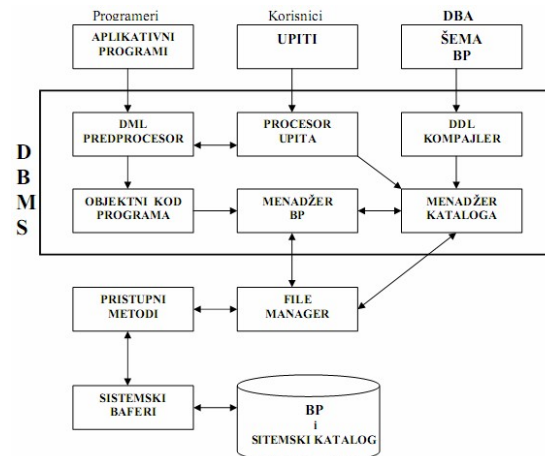
- Memorisanje, pretraživanje i ažuriranje podataka – osnovna funkcija
- Katalog dostupan korisniku
- Podrška Transakcijama – sprovode se sva ažuriranja ili se nijedno ne ostvaruje
- Upravljanje konkurentnim pristupom – obezbeđuje korektno ažuriranje BP kada veći broj korisnika istovremeno ažurira bazu podataka
- Servisi Oporavka (Recovery)
- Servisi autorizacije – obezbeđuje da samo autorizovani korisnici imaju pristup BP.
- Servisi integriteta – su mehanizmi zahvaljujući kojima i podaci i njihove promene moraju postovati određena pravila.
- Podrška komunikaciji
- Uslužni servisi

15. DBMS-osnovne komponente

Komponente DBMS–a:

- DDL kompajler vrši konverziju DDL instrukcija u skup tabela koje sadrže meta–podatke. Te tabele se memorišu u Sistemskom Katalogu.
- Menadžer kataloga upravlja pristupom i održava sistemski katalog. Preko njega pristupa većina komponenata DBMS–a.
- Procesor Upita je komponenta koja transformiše upite u seriju instrukcija niskog nivoa koje se usmeravaju ka DB Menadžeru.
- DML predprocesor konvertuje DML instrukcije, ugnježdene u apl. programu, u standardne funkcijske pozive u sklopu Host jezika.

- Menadžer BP ima interfejs prema aplikacijama i upitima korisnika. On prihvata upite, ispituje Spoljnu i Konceptualnu šemu kako bi odredio koji su rekordi potrebni. Potom generiše poziv Fajl Menadžeru kako bi ovaj obavio zahtev.
- Fajl Menadžer manipuliše memorijskim fajlovima na nižem nivou i upravlja alokacijom memorijskog prostora na disku. Uspostavlja i održava Listu Struktura i Indeksa koji su definisani u Internoj šemi. FM ne upravlja direktno fizickim U/I podataka nego prosledjuje zahteve za odgovarajucim Pristupnim metodama koji ih U/I u Sistemski Bafer ili Keš (Cache) bafer.



16. DML funkcije

DML predprocesor konvertuje DML instrukcije, ugnježdene u apl. programu, u standardne funkcijske pozive u sklopu Host jezika.

17. DDL kompajler

DDL kompajler vrši konverziju DDL instrukcija u skup tabela koje sadrže meta-podatke. Te tabele se memorišu u Sistemskom Katalogu.

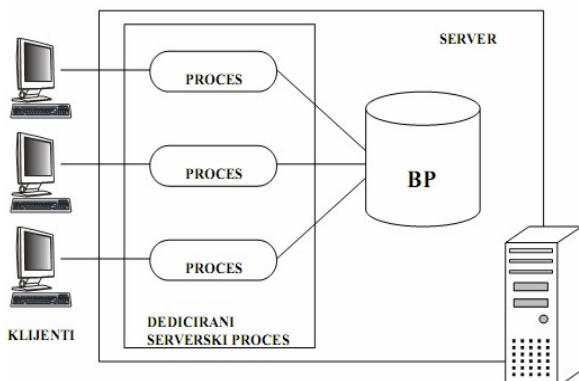
18. Funkcije SQL DB servera

- SQL DB server je mešavina standarnde SQL funkcionalnosti proširenja specifičnih za datog proizvođača
- Do 1998 se svasta uključivalo u SQL server, danas su to obično posebni produkti, npr. TPM, Web, Aplikativni serveri, ... (TPM–Transaction Processing Monitor)
- On upravlja kontrolom i izvršavanjem SQL komandi. Obezbeđuje logičke i fizičke poglede na podatke i generiše optimizovane planove pristupa za izvršavanje SQL komandi.
- Od DB-servera C/S aplikacija zahteva podatke i sa njima povezane servise (sort,...).
- DB-server se ponekad naziva “SQL-engine” obezbeđuje siguran pristup deljenim podacima i odgovara na zahteve klijenata.
- Kako SQL server omogućava većem broju apl.da pristupaju istoj BP u isto vreme, on mora obezbediti okruženje koje štiti BP, tj. Upravlja oporavkom, konkurentnim pristupom, sigurnošću, integritetom, kontrolom transakcija, zaključivanjem.
- Kao min. Vecina SQL servera obezbeđuje funkcionalnost na nivou SQL’89. Vecina uključuje i neka svojstva SQL’92. Samo nekolicina obezbeđuje proizvodacke verzije mem. procedura , triggera i pravila (SQL–3)

19. Arhitektura SQL DB servera

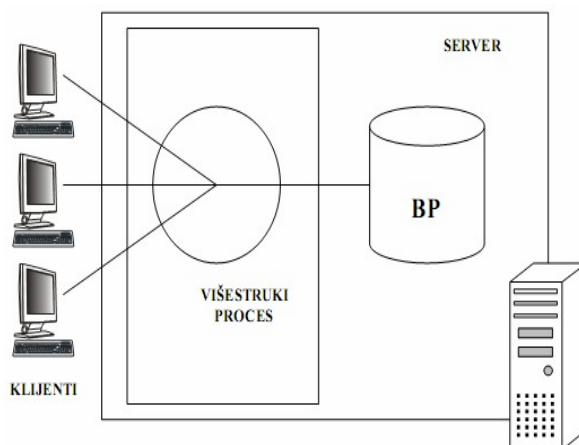
Danas se koriste 3 razlicite arh. servera koje DB koriste za prihvatanje udaljenih klijenata BP i to :

- Arhitektura tipa proces-po-klijentu



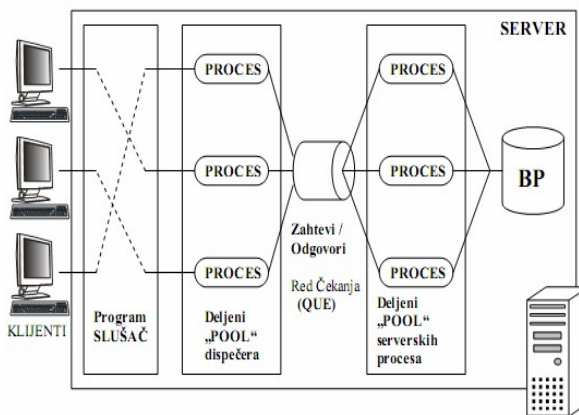
- Obezbedjuje maksimalnu sigurnost dajuci svakom klijentu BP sopstveni adresni prostor. BP radi u 1 ili više odvojenih pozadinskih procesa (Informix, DB2, Oracle 6)
- Prednosti: štite korisnika jedan od drugog. Štiti DB menagera od korisnika. Procesi mogu lako biti pridruženi razlicitim procesorima (SMP). Za svoje multitasking servise arhitektura je oslonjena na lokalni OS.
- Nedostaci: zahteva više memorije i CPU resursa nego ostale, Može biti
- Sporija zbog cestih promena konteksta procesa (Prevazilazi se TPM).

- Arhitektura sa više niti.



- Obezbedjuje dobre performanse izvršavajući sve konekcije korisnika, aplikacije i BP u istom adresnom prostoru.
- Ova arh. ima svoj sopstveni interni Scheduler, tj. ne koristi šeme lok. OS za raspodeljivanje zadataka i adresnu zaštitu (Sybase, MS SQL Server)
- Prednosti: cuva memoriju i CPU resursem pošto ne zahteva ceste promene konteksta. Serverske implementacije su i nešto portabilnije.
- Nedostaci: Korisnicka apl. koja se „loše“ ponaša može „oboriti“ ceo DB server i njegove taskove. Korisnicke apl. koje se sastoje od taskova dugog trajanja (dugacki upiti) mogu zauzeti resurse servera. „Preemptive“ rasporedivac koji obezbeduje DB server je po pravilu slabiji od schedulera nativnog OS.

- Hibridna arhitektura se sastoji od 3 kljucna elementa:



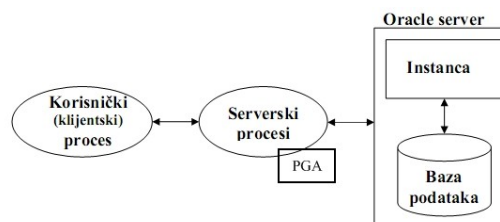
- Više nitne mreže „slušaca“ koji ostvaruju pocetnu konekciju tako što klijenta pridružuju dispeceru
- Dispecerskih taskova koji poruke smeštaju u interni red poruka (message que) i iz njega šalju odgovore klientima
- Severskih, deljenih, radnih procesa, koji se mogu ponovo koristiti, koji uzimaju posao iz reda cekanja, izvršavaju ga i smeštaju u izlazni Que (Oracle 7, Oracle 8)
- Prednosti: Ove arhitekture obezbeduju zašticeno okruženje za izvršavanje korisnickih taskova, ali bez pridruživanja stalnih procesa svakom klijentu.
- Nedostaci: Kašnjenja u redu cekanja.

20. Izbor arhitekture SQL DB servera

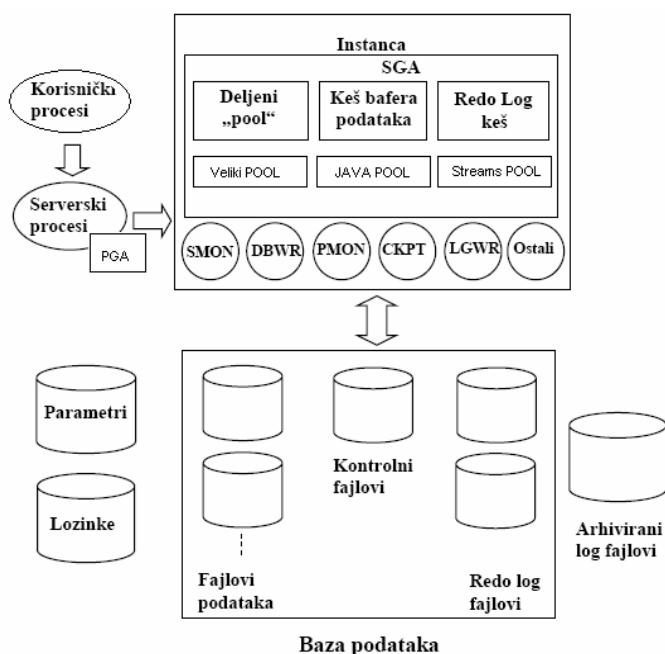
- Kada ima puno korisnika povezanih na BP, arh. sa procesom po klijentu može imati loše performanse ali ona obezbeđuje najbolju zaštitu.
- Visenitne arh. obezbeđuju dobar rad velikog broja korisnika koji generišu kratke transakcije, Ova arh. ne obezbeđuje potpunu (Visoku) sigurnost.
- Hibridne arh. nude najbolji balans između sigurnosti i performansi.
- Stvar nije kritična za jednostavan LAN– bazirani DSS, ali može biti za ozbiljan transakcioni (OLTP) sistem (*OLTP–On-Line Transaction Processing*)
- Ostala važna svojstva DB (SQL) servera
 - Memorisane procedure (store proc.)
 - Trigeri(okidaci)
 - Pravila

21. Arhitektura RDBMS Oracle10g

- Oracle server predstavlja OR DBMS nastao objektnim proširenjem relacionog modela koji obezbeđuje otvoreni i integrisani pristup upravljanju informacijama.
- Oracle server se sastoji od BP i instance (procesa i sistemske memorije na serveru koji obezbeđuju pristup BP) i mogu se konektovati samo na jednu BP.
- Korisnički proces predstavlja apl. program korisnika u sklopu koga se generisu SQL instrukcije.
- Serverski proces izvršava SQL instrukcije dobijene od korisničkog procesa.



22. Detaljna arhitektura Oracle10g servera



23. Oracle10g procesi

Postoje serverski procesi koji upravljaju zahtevima koji dolaze od konektovanih korisnickih procesa i pozadinski procesi, koji obavljaju asinhroni I/O. Ovi procesi mogu biti obavezni i opcionalni. Svaka instanca mora ukljuci sledece obavezne pozad procese:

- **DBWR** (*Database writer*) - ima zadatak da modifikovane blokove podataka iz SGA bafer keša upide u fajlove podataka na disku. Omogucava odlaganje upisivanja do pojave spec. uslova. Instanca može imati do 10 DBWR procesa.
- **LGWR** (*Log writer*) - ima zadatak da prepisuje podatke iz SGA Redo log bafera u aktivni Redo log fajl na disku. Obavlja se sekvencijalno, za slucaj pojave spec. uslova. Ovaj proces potvrđuje commit samo posle upisa redo informacije na disk.
- **CKPT** (*Checkpoint process*) - ima zadatak da ažurira informaciju o statusu BP u upravljackim i data fajlovima svaki put kada se promene u Keš baferima permanentno registruju u fajlovima BP, CKPT proces se koristi za sinhronizaciju fajlova BP.
- **SMON** (*System monitor*) - ima zadatak da proverava konzistentnost BP, i u slucaju potrebe, inicira oporavak BP. SMON vrši automatski oporavak instance u slucaju njenog kvara.
- **PMON** (*Process monitor*) - ima zadatak da prati korisnicke procese koji pristupaju bazi i da ih oporavlja posle eventualnog pada.
- **PGA** (*Program Global Area*) - Globalno podrucje u memoriji koje se dodeljuje serverskom procesu.

24. Oracle instanca

- Predstavlja kombinaciju memorijskih struktura i pozadinskih procesa.
- Da bise pristupilo podacima u bazi instanca mora biti pokrenuta, kojom prilikom se dodeljuje podrucje zajednicke memorije (SGA –System Global Area) i pokrecu pozadinski procesi.
- U jednom momentu instancu može otvarati i koristiti samo jedna BP.
- SGA se koristi za memorisanje onih podataka iz baze koji se zajednicki koriste od strane procesa BP. Pozadinski procesi obavljaju funkcije u ime pozivajućeg procesa. Obavljaju I/O i nadziru druge procese.
- SGA podrucje je sastavljeno od nekoliko memorijskih struktura:
 - **Shared pool** (*zajednicke deljene zalihe*) - memorise skoro koriscene SQL naredbe, kao i skoro koriscene podatke iz recnika podataka.
 - **Data buffer cache** (*keš bafer BP*) - koristi se za memorisanje skoro korišćenih podataka
 - **Redo Log buffer cache** (*Keš bafer dnevnika izmena*) - koristi se za pracenje promena koje u bazi posredstvom instance izvedu serverski i pozadinski procesi. Ovo se koristi u slucaju potrebe oporavka baze podatka.
 - **Large POOL** (*velike zalihe*) - Opcionalno memorijsko podrucje koje služi za smanjenje opterećenja kojem je izlozen zajednicki pool (zalihe).
 - **Java POOL** (*Java zalihe*) - Opcionalna memorijska struktura - nuzna u slucaju da se koristi Java
 - **Streams POOL** (*zalihe tokova*) - Koristi se u sklopu infrastrukture za asinhrono deljenje podataka

25. Logička struktura Oracle BP

Na logickom nivou Oracle operise sledecim pojmovima:

- **Prostor tabela** (*table space*) - Oracle BP je podeljena na memorijske logicke jedinice npr. koje se zovu prostor tabela. Svaka Oracle BP sadrži Table space koji se zove SYSTEM, koji se automatski formira i sadrži tabele Sistemskog kataloga za celokupnu BP. Dobro je da postoji još jedan Table Space za korisnicke podatke.

- Korisnik (*user*) predstavlja ime definisano u BP koji se na nju može konektovati i koji može pristupiti objektima (podacima).
- Šema predstavlja imenovanu kolekciju objekata šeme (tabele, pogledi, klasteri i procedure) pridruženu određenom korisniku.
- Blok podataka predstavlja najmanju jedinicu memorije koju Oracle može koristiti ili dodeliti. Definiše se prilikom kreiranja BP. Velicina ovog bloka treba da bude multipl velicine bloka u sklopu OS.
- Ekstent predstavlja određeni broj susednih blokova podataka.
- Segment predstavlja skup ekstenata dodeljenih određenoj logičkoj strukturi (npr. Podaci svake tabele se pamte u svom posebnom segmentu podataka, dok se podaci o Indexima u svojim sopstvenim index-segmentima).

26. Fizička struktura Oracle BP

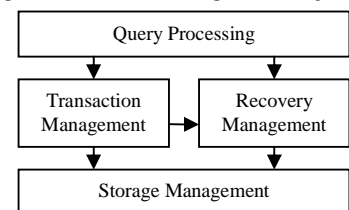
Oracle BP se sastoji od 3 tipa podataka:

- Data files (*Fajlovi podataka*) - sadrže stvarne podatke u BP.
- Redo Log files (*Fajlovi sa listom akcija za obnavljanje*) - registruju sve promene izvedene nad podacima. Svaka Oracle BP raspolaze sa najmanje dva Fajla sa listom akcija za obnavljanje (redo log).
- Control files (*Upravljacki fajlovi*) - sadrže listu svih fajlova koji cine BP, sa ciljem održavanja i verifikacije integriteta BP. Baza zahteva postojanje barem 1 kontrolnog fajla.

27. Arhitektura RDBMS My SQL

My SQL aplikacioni sloj je tamo gde korisnici medjusobno komuniciraju sa MySQL RDBMS.

- Query Processor (*procesor upita*) – Velika kolicina uzajamnog dejstva u sistemu se događa kada korisnik zeli da vidi ili manipulise osnovnim podacima u skladistu.
- Transaction Management (*upravljanje transakcijama*) – ima odgovornost da osigura da je transakcija registrovana i izvršena automatski.
- Recovery Management (*upravljanje oporavkom*) – je odgovoran za registrovanje svake izvršene operacije u BP. Isto, odgovoran je za obnavljanje BP u njeno poslednje stabilno stanje.
- Storage Management – Upravljanje skladistenjem.



MySQL konceptualna arhitektura visokog logickog nivoa

28. Objektno-Relacioni DBMS (OR-DBMS)

- Pokušaj ujedinjenja Relacionih i Objektnih BP.
- Koristi model podataka koji BP dodaje objektnu orijentisanost.
- Sva perzistentna informacija je u tabelama, ali neki elementi tabela imaju bogatiju strukturu, ADT (Abstract Data Types): Tekst, Slika, Grafika, Animacija, Audio, Video, georeferencirani, Vremenski markirani...
- Ne postoji Enkapsulacija metoda sa podacima, ogranicena podrška drugim OO svojstvima.
- Podržavaju prošireni oblik SQL-a (Object SQL, upiti sa ugnjeđenim Objektima, atributi sa skupom vrednosti, uključivanje metoda i iskaza pretraživanja koji uključuju i elemente ADT).

- Nedostaje direktna podrška OO Jezicima i njihovim objektima, nego se mora "PREVODITI" između objekata i tabela.
- Primeri: Sybase Enterprise Aplicatin Server, Informix Universal Server (Illusra), IBM Universal DB (DB2 Extenders), MS Repository, Oracle Universal Server (Oracle 8)

29. Objektno-Orijentisani DBMS (OO-DBMS)

- Ne postoji zvaničan standard za Objektno BP, de-facto standard je ODMG v.2.0 definisan od strane Object Database Management Group.
- Objektno BP koriste model podataka koji ima OO koncepte– klasa koja obezbeđuje identifikatore objekata (OID) za svaku perzistentnu instancu klase, enkapsulaciju, višestruko nasledjivanje i podržava ADT.
- Objektno orjentisani jezik predstavlja Jezik kako za aplikaciju tako i za BP.
- ODBMS je do sada integrisan sa: C++, C, Smalltalk, Java...
- Deklarativni jezik za upit DB objekata je OQL (Object Query Language) i on nije 100% kompatibilan sa SQL–om. Vecina OODBMS podržava i SQL putem ODBC–a (Open Data Base Conectivity).
- Svakom objektu se automatski dodeljuje jedinstven, nepromenljiv identifikator–OID.
- Podesno za CAD/CAM sisteme, danas za finansijske, telekomunikacione i www aplikacije.
- Primeri: Gemstone, jasmine(CAI), Itasca (Ibex obj. sys), Poet v.5.0, O2 (Ardent s/w), Objectivity/DB, Ontos DB,...

30. Modeli podataka

31. SQL i programski interfejsi-Evolucija

SQL (Structural Query Language) spada u kategoriju neproceduralnih, deklarativnih jezika i ima dve glavne komponente:

- DDL (Data Definition Language) – jezik za definisanje strukture BP i kontrolu pristupa podacima.
- DML (Data Manipulation Language) – jezika za pretrazivanje i azuriranje podataka

Evolucije SQL-a:

- SQL-89
 - ANSI/ISO standard
 - postojanje referencijalnog integriteta
 - Standardizacija ugnježdjenog SQL-a (samo za Fortran, COBOL, PL/I, Pascal).
- SQL-92
 - Finija kontrola transakcija
 - Standardizovani katalog
 - Podrška ugnježđenom SQL (za C, Ada, MUMPS)
 - Podrška novim tipovima podataka (BLOB, VARCHAR, TIME, DATE, TIMESTAP)
 - Podrška JOIN operatoru
- SQL-99
 - Object SQL
 - Memorisanje procedure

- Trigeri
- Korisnicki definisani tipovi (UDT)
- CLI – interfejsi na nivou poziva
- Multimedija

32. Programski interfejsi BP

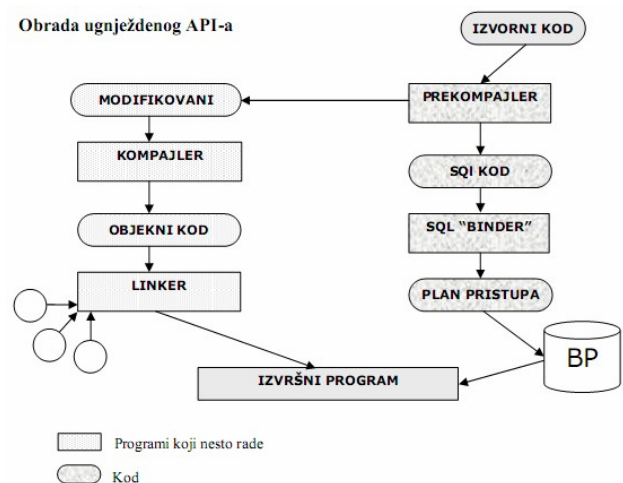
- Programski interfejsi BP (DB API i SQL API) ispunjavaju sledece funkcije:
 - Prihvataju zahteve aplikacija u pogledu podataka
 - Te zahteve prosledjuju do DBMS-a
 - Rezultat vraćaju do aplikacije
- Postoje dva osnovna tipa DB-API-a:
 - Ugnježdjeni API (EAPI=Embedded Application Programming Interface - API)
 - API na nivou poziva (CLI-API=Call Level Interface API)

33. Ugnježdjeni API (ESQL API), funkcije, obrada

- SQL komande su ugnježdjene u tradicionalni programski jezik (2 ili 3 gen.) koji daju upravljačka svojstva (DO WHILE, GO TO).
- Prekompilacijom se SQL iskazi odvajaju od klasa programa i odvojeno se obrađuju.
- Nacin pristupa BP je fiksiran i “kodiran” u programu.
- Staticke SQL instrukcije
- E-API može biti:
 - Staticki ugnježdjeni SQL
 - Dinamicki ugnježdjeni SQL

34. Obrada ugnježdjenog SQL API-ja

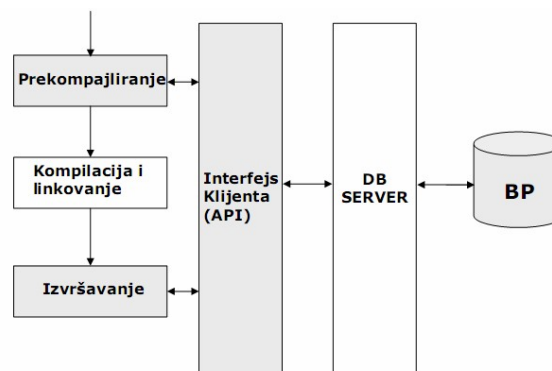
- Izvorni kod se propusta kroz prekompajler i razbija na modifikovani kod i SQL kod. Modifikovani kod je upravo C-kod koji se propusta kroz C-kompajler, a zatim kroz linker, nakon čega imamo izvršnu verziju koda. Sa druge strane, deo originalnog koda koji je izdvojen kao SQL kod se prolaskom kroz SQL binder konvertuje u plan pristupa BP. Rano povezivanje se obavlja prilikom prekompajliranja.
- Nedostaci: U toku razvoja aplikacije baza podataka mora biti poznata i raspoloživa. Mada nefleksibilan, ESQL DB-API nudi najbolje performanse.



35. DB interfejsi na nivou poziva (CLI API)

- Konceptualno prostiji, zbog koriscenja biblioteke funkcija.
- Koristi dinamicke SQL instrukcije koje se kreiraju u toku izvršavanja. Ovaj metod se naziva kasno povezivanje jer se SQL komande zadaju u trenutku izvršavanja, te ne postoji potreba da BP bude ranije poznata, a nisu potrebni ni prekompilacija i povezivanje.

- Tipična sekvenca:
 - Uspostavljanje veze sa BP
 - Priprema (buffera) SQL zahteva
 - Izvršavanje zahteva (obrada buffera)
 - Obrada statusa i rezultati
- CLI-API može biti:
 - Proizvodjaci (nativni)
 - Standardni CLI-API



36. Komparacija ESQL API i CLI API

- EAPI zahteva da ciljna baza bude unapred poznata, a CLI-API ne.
- EAPI podržava prirodno statički SQL, a CLI-API kroz sored procedure.
- Oba podržavaju dinamički SQL (CLI-API prirodno)
- EAPI zahteva prekompliaciju i povezivanje (binding), a CLI-API ne.
- EAPI je lakši za programiranje od CLI-API-a.
- CLI-API je lakše debugovati i pakovati i više je tool-frendly.

37. Osnovi upravljanja procesima

- S obzirom na dinamičku prirodu podataka razmatra se način izvođenja kompleksnih izmena nad podacima uz održavanje njihove konsistentnosti.
- Osnovu upravljanja procesima predstavlja koncept transakcije.
- Upravljanje procesima podrazumeva:
 - Upravljanje transakcijama
 - Upravljanje konkurentim pristupom
 - Kontrola sigurnosti
 - Problemi sa procesnom transparentnošću.

38. Upravljanje transakcijama

- Transakcija (UOW – unit of work) predstavlja jednu ili više akcija nad BP koje se tretiraju kao celovita jedinica posla. Ili se izvršavaju sve akcije jedne transakcije, ili se ne izvršava ni jedna.
- Ukoliko je transakcija uspešna, za nju se kaže da je izvršena (commit).
- Ukoliko transakcija nije uspešna, za transakciju se kaže da je vraćena na početak (rolled back) ili napuštena (aborted).
- Postoje tri komande koje se mogu izvršiti nad transakcijama:
 - BEGIN TRANSACTION – označava početak transakcije
 - COMMIT - promene u BP čini trajnim
 - ABORT/ROLLBACK - poništava izmene koje je napravila tekuća transakcija

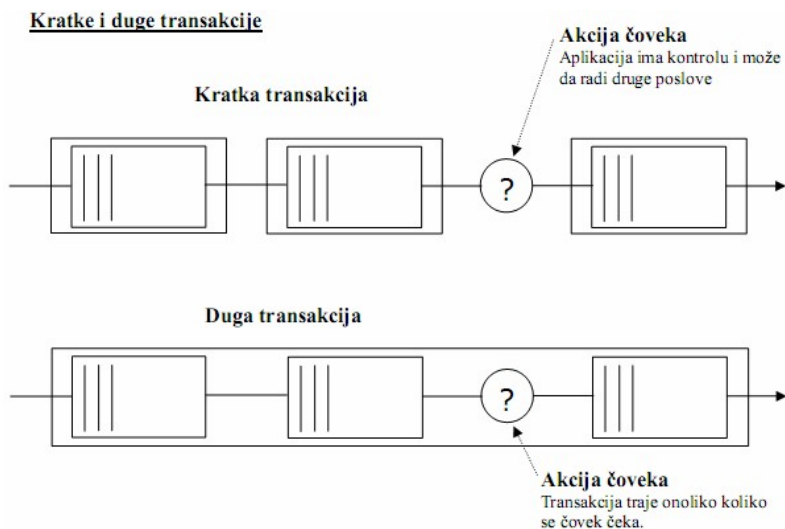
39. Svojstvo ACID

- Atomarnost (Atomicity) - zahtev za nedeljivu jedinicu posla.

- Konsistentnost (Consistency) - zahtev da aplikacija pomera BP iz jednog u drugo konsistentno stanje. Nakon komitovane transakcije, stanje u bazi mora biti konsistentno, u suprotnom ce se transakcija odbiti.
- Izolacija (Isolation) - zahtev da jedna transakcija ne može uticati na druge transakcije.
- Trajnost (Durablity) - zahtev da su efekti transakcije nakon komitovanja trajni.

40. Kratke i duge transakcije, definicija i komparacija

- Kratke transakcije su reda nekoliko delova sekunde, nekoliko sekundi ili par minuta. Odlikuje ih paketna obrada, dobra efikasnost, ali fleksibilnost interakcije korisnika slabija.
- Duge transakcije mogu trajati satima, pa cak i danima. One imaju "konevuzionu prirodu", odnosno neophodna je interakcija sa korisnikom pa je i fleksibilnost interakcije korisnika dobra, ali je efikasnost slabija. Primer za duge transakcije je pretrazivanje svih racunara u nekoj banci.

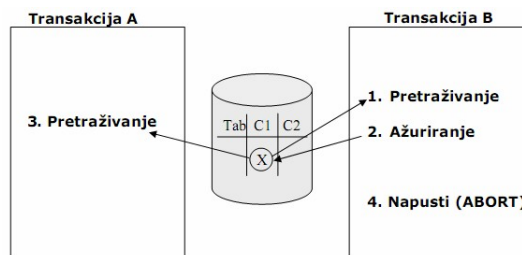


41. Upavljanje konkurentnim pristupom

Konkurentan pristup podrazumeva da se više transakcija konkurentno izvršava nad BP. Kada imamo konkurentan pristup istom resursu BP, moguća je pojava niza problema:

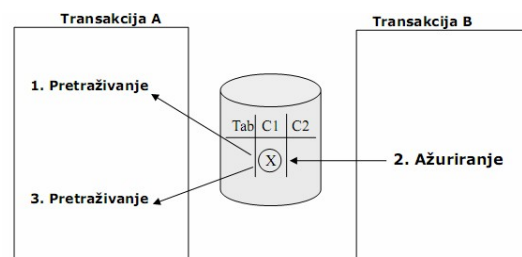
- Necista (prljava) citanja

U slučaju da commit izostane (abort transakcije), druga strana (TA) ima netacan podatak jer uzima vrednost koja je samo privremeno promenjena.



- Neponovljiva citanja

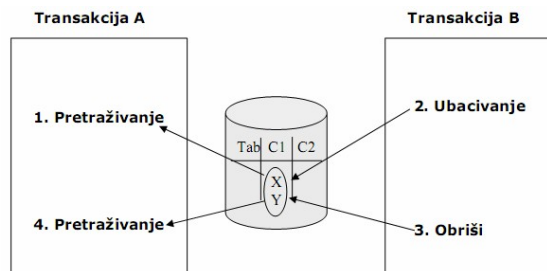
Desava se kada se redovi ne zaključavaju za akciju citanja (pretraga) pa dolazi do toga da isto polje ima različite vrednosti. Tada strana koja pretražuje može videti izmenjene podatke pre commit-a TB.



- **Fantomski redovi**

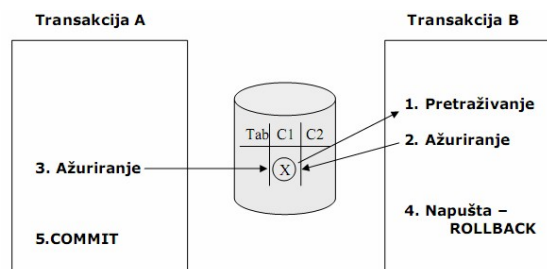
TA izvršava dva identična upita, a rezultat je drugaciji, tj. za isti upit neki redovi su nestali, a neki su novi.

Desava se kada nije dodeljen obuhvat lock-ova prilikom select-a za TA.



- **Izgubljena ažuriranja**

Desava se ukoliko dodje do preklapanja operacija pre commit-a obe transakcije. U ovom slučaju ažuriranje TB je izgubljeno, iako se uspesno završilo, tj. ne može da se vrati u prvobitno stanje jer je druga transakcija (TA) vrsila ažuriranje.



42. Problemi konkurentnosti (transakcija)

- Serijsko izvršavanje transakcija (kada se transakcije ne poklapaju) nikada neće dovesti do ovih problema, sistem je uvek konsistentan.
- Uporedno izvršavanje skupa transakcija je serijabilno, ako proizvodi isti efekat u BP kao i neko serijsko izvršavanje istog skupa transakcija.
- Primeri protokola za ostvarivanje serijabilnosti izvršenja skupa transakcija su:
 - Pesimistički pristup
 - Optimistički pristup
 - TPM

43. Pesimistički pristup upravljanju konkurentnim radom

- Pristup je pesimistički jer polazi od stanovišta da će do koalicije sigurno doći. Funkcionise tako što se na referisan objekat BP postavlja Lock (katanac) pre njegovog koriscenja.
- Tehnika je efikasna ali su performanse slabije.
- Moguća je pojava dead-lock-ova.
- Moguće je poboljšanje pesimističkog pristupa, uvođenjem:
 - Tipova zaključavanja
 - Granularnost zaključavanja (nivo objekata BP na kome se vrši zaključavanje, cela BP, tabela, stranica, red, kolona)
 - Nivoi izolacije (potpuna i delimična)
 - Trajanje zaključavanja
 - Detekcija i razrešavanje dead-lock-ova

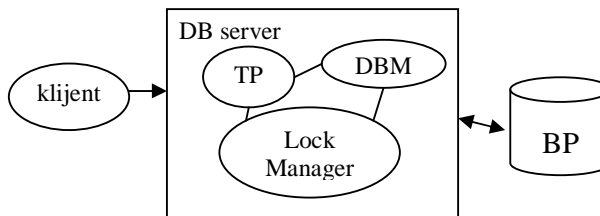
44. Nivoi zaključavanja objekata BP

Što je zaključavanje na nižem nivou, veću je moguću paralelizam i bolje su performanse, i obrnuto. Nivoi zaključavanja objekata baze mogu biti:

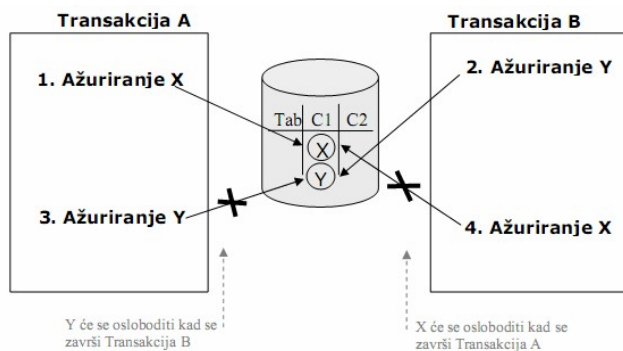
- Nivo cele baze
- Nivo tabele
- Nivo stranice (bloka)
- Nivo reda
- Nivo kolone

45. Upravljanje transakcijama primenom TPM

- TPM spada u kategoriju middleware-a, a služi za upravljanje transakcionim radom.
- Klijentski zahtevi stizu prvo do TPM-a, koji onda konsultuje lock manager-a i u skladu sa rezultatom (trenutnim stanjem trazenih objekata BP) predaje odabrane zahteve DBMS-u.
- TPM obezbedjuje da svi sistemi povezani sa transakcijom ostanu u konsistentnom stanju posle njenog zavrsetka.



46. Pojava i razrešavanje dead-lock-ova



TA i TB se međusobno sprečavaju da zavrse akcije, sto prouzrokuje deadlock, Sistem se brani tako sto jednu transakciju odbaci (abort), da bi se druga zavrсила.

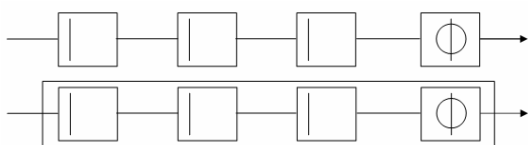
47. Upravljanje sigurnošću baze podataka

- DBA može davati dozvole pomoću GRANT/REVOKE privilegija. Dizvole koje se korisniku mogu dati obuhvataju:
 - Sistemske privilegije (npr. Create Table)
 - Privilegije nad objektima (Select, Insert, Update, Delete)
- Umesto pojedinacnog dodeljivanja privilegija korisniku moguće je dodeljivanje *role* koja predstavlja imenovani skup privilegija.
- Definicija pogleda (view) takođe pomaže u upravljanju sigurnošću BP. Pomoću pogleda obezbedjuje se različit prikaz različitim korisnicima nad istim podacima. Pogled je objekat BP koji za razliku od tabele ne sadrži same podatke.

48. Problemi sa transparentnošću procesa

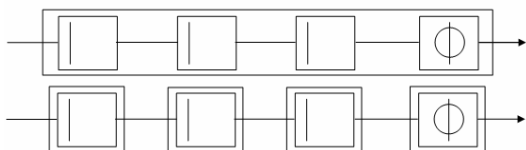
Posledica nesklada dva okruženja u kojima se izvršava ista transakcija u pogledu:

- Nesklad granica transakcije
- Nesklad izolacionih nivoa
- Nesklad u privilegijama i identifikaciji korisnika



Prva transakcija projektovana je sa Auto-Commit-om posle svake faze pa se usled neuspeha neke faze ne vraća na pocetak transakcije, nego samo jedan korak unazad.

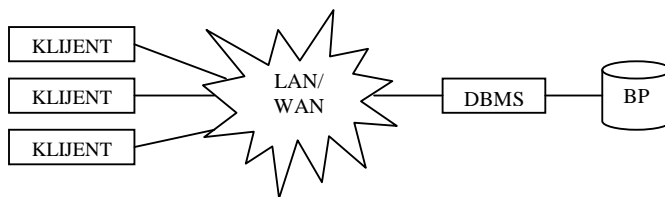
Transakcija 2 je identicna prvoj, ali se odvija u drugacijem okruzenju, bez opcije autocommit, te ce se ona u slucaju neuspeha vratiti na pocetak transakcije.



Situacija simetrična prethodnoj. Transakcija 2 je inicijalno projektovana bez Auto Commit-a posle svakog koraka, ali kada radi u drugacijem okruženju, drugacije se ponaša, tj vršice se Auto-Commit posle svakog koraka.

49. Distribuirani sistemi (definicije, prednosti, nedostaci)

- Distribuirana obrada predstavlja izvršavanje kooperativnih procesa koji međusobno komuniciraju putem razmene poruka preko (informacione) mreže. Realizuje se kao centralizovana baza podataka kojoj se se može pristupiti preko racunarske mreže.
- Distribuirana BP je logicki integrisani skup deljenih (zajednicki korišćenih) podataka koji su fizicki distribuirani na više servera koji su povezani mrežom.
- Distribuirani DBMS (DDBMS) se definiše kao s/w za upravljanje distribuiranom bazom podataka, na takav nacin da su aspekti distribucije transparentni za korisnika.
- Kod distribuiranih sistema javljaju se brojne komplikacije npr. konkurentne transakcije, održavanje konsistentnosti, kesiranje i repliciranje, održavanje konsistentnosti kesa i replika...



50. Distribuirana obrada, distribuirana baza i distribuirani DBMS

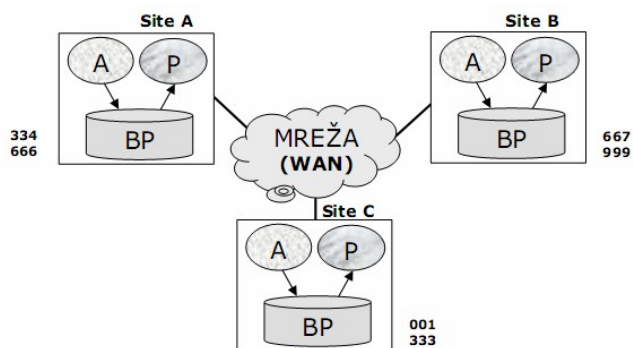
- Distribuirana obrada predstavlja izvršavanje kooperativnih procesa koji međusobno komuniciraju putem razmene poruka preko (informacione) mreže. Realizuje se kao centralizovana baza podataka kojoj se se može pristupiti preko racunarske mreže.
- Distribuirana BP je logicki integrisani skup deljenih (zajednicki korišćenih) podataka koji su fizicki distribuirani na više servera koji su povezani mrežom.
- Distribuirani DBMS (DDBMS) se definiše kao s/w za upravljanje distribuiranom bazom podataka, na takav nacin da su aspekti distribucije transparentni za korisnika.

51. Tehnike distribucije podataka

- Partitioniranje - Tabela se deli po horizontali ili po vertikali u manje delove (fragmente) bez preklapanja, koji se distribuiraju na 2 ili više servera.
- Ekstrakcija – predstavlja kopiju dela baze u jednom trenutku.
- Kesiranje - Predstavlja tehniku distribucije podataka gde se pravi privremena (dinamicka) replika BP ili njenog dela.
- Replikacija – predstavlja kopiju master BP koja se može ažurirati.

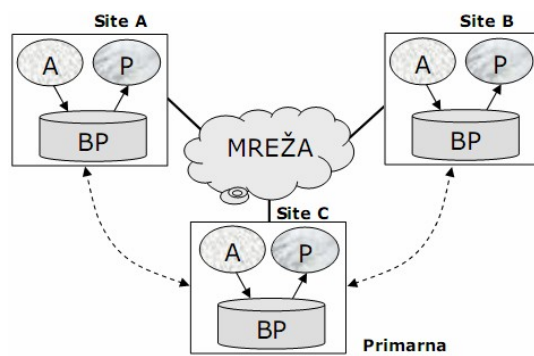
52. Horizontalno i vertikalno particioniranje podataka iz BP

- Particioniranje - Tabela se deli po horizontalni ili po vertikalni u manje delove (fragmente) bez preklapanja, koji se distribuiraju na 2 ili više servera.
- Tehnika je efikasna i sigurna, obezbedjuje paralelizam i pruza pogodno nacine koriscenja BP putem pogleda.
- Performanse su slabije.



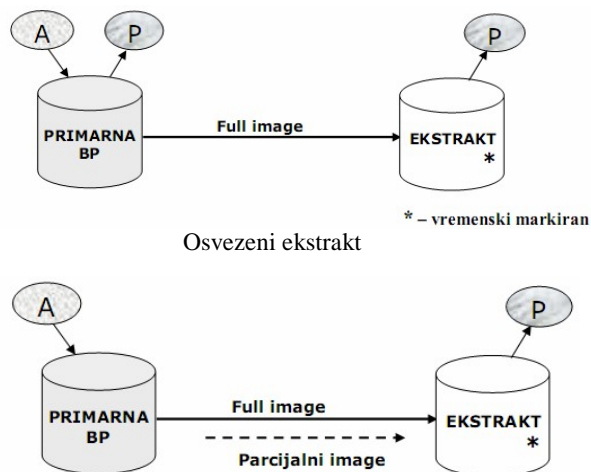
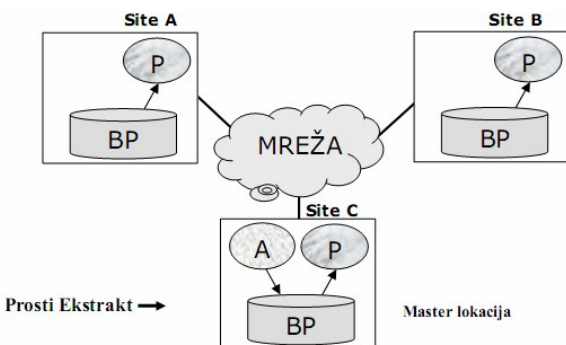
53. Keširanje podataka

- Keširanje - Predstavlja tehniku distribucije podataka gde se pravi privremena (dinamicka) replika BP ili njenog dela (na serveru ili klijentu ili na oba).
- Poboljšava performanse rada ako se jednom referisani podaci cesto koriste (npr. Internet) i ako se podaci retko menjaju.
- Problem je održavanje konsistentnosti keša.

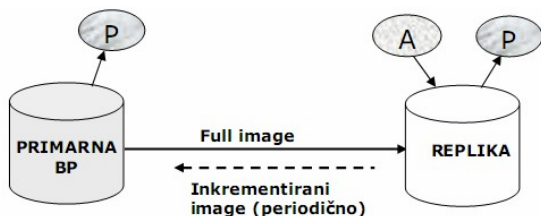


54. Ekstrakt i replika

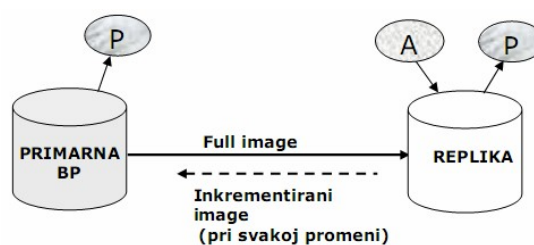
- **Ekstrakcija** – predstavlja kopiju dela baze u jednom trenutku.
- Prosta i jeftina tehnika distribucije podataka zasnovana na kopiranju.
- Ekstrakt se neažurira, ali se ponavlja, tj. vrši se pravljenje novog ekstrakta nad delom BP.
- Ima smisla ako su podaci nepromenljivi ili sporo promenljivi npr. za istorijske, arhivske podatke.
- Postoji razni tipovi ekstrakta:
 - Prosti ekstrakt
 - Vremenski markirani ekstrakt
 - Osveženi ekstrakt



- **Replikacija** – predstavlja kopiju master BP koja se može ažurirati.
- Periodično se vrši sinhronizacija master BP slanjem inkrementiranog imidža sa replike.
- Danas veoma često korišćena tehnika distribucije podataka.
- Postoje dva tipa shodno načinu sinhronizacije sa BP:
 - Periodična replika
 - Kontinualna replika



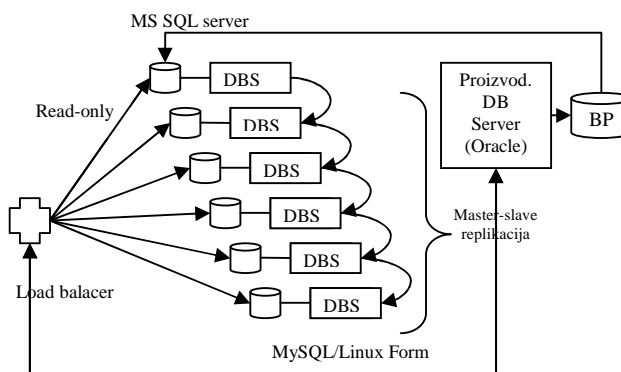
Primarna BP se ne azurira, već se azuriraju replike, a potom se vrši sinhronizacija sa BP periodično.



Prilikom svakog azuriranja replike šalje se novi image zbog sinhronizacije sa BP

55. Replika kod MySQL-a

Replikacija omogućava podacima sa MySQL master servera BP (glavni server) da budu replicirani na jedan ili više slave servera BP (sporednih servera). Replikacija je asinhrona - nema potrebe da stalno bude povezivana na primanje azuriranja od master-a, čime se misli da se azuriranje može dogoditi preko udaljene konekcije, čak i preko privremenog resenja kao što je dial-up servis. U zavisnosti od konfiguracije, mogu se replicirati sve BP, odabrana BP pa čak i selektoavna tabela unutar BP. Koriste se forme open-source DB servera, pre svega za “read-mostly” aplikacije (75% citanje, 25% upis), sa lansiranom master-slave replikacijom.

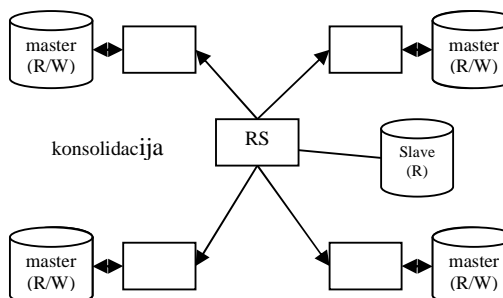
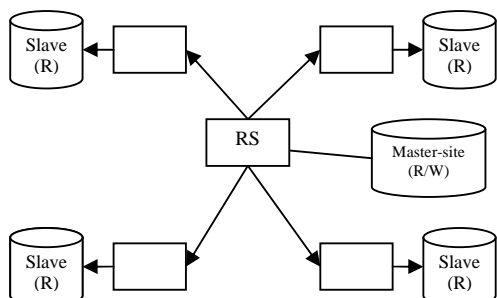


56. Korišćenje replikacionih servera

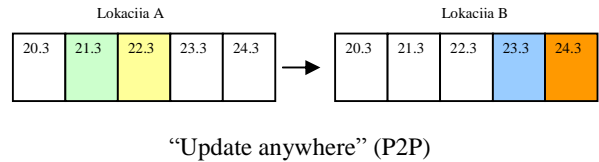
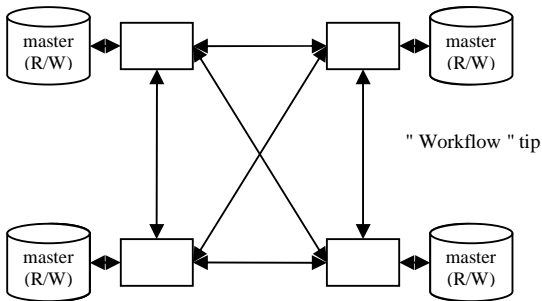
67. Vlasništvo nad podacima

Postoji nekoliko načina korišćenja replikacionih servera, zavisno od:

- Trenutka azuriranja replike
 - Sinhrona replikacija – podaci se azuriraju na replici odmah posle azuriranja izvorne BP.
 - Asinhrona replikacija – ciljna baza se azurira sa zadrskom.
- Funkcionalnosti
- **Vlasništva nad podacima**
 - **Master-slave** – asinhrono replicirani podaci kod ovog tipa su vlasništvo jedne lokacije i mogu biti azurirani samo od strane te lokacije (Master-slave diseminacija i konsolidacija).



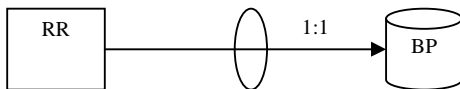
- **Workflow** – pravo azuriranja repliciranih podataka se prenosi sa sajta m sakt.
- **Update anywhere (P2P)** – svi sajtovi predstavljaju master lokacije, i kada se na nekom od njih izvrši azuriranje, svi ostali sajtovi u mrezi se obavestavaju.



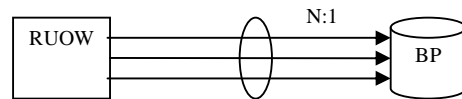
57. Načini distribucije procesa

- Moguće različite klasifikacije transakcija, a ovde ćemo navesti IBM DRDA (*Distributed Relational Database Architecture*) bazirana. (Zahtev = *SQL instrukcija*, *UOW* = transakcija)
- Četiri načina distribucije procesa, sa rastućim nivoom kompleksnosti:

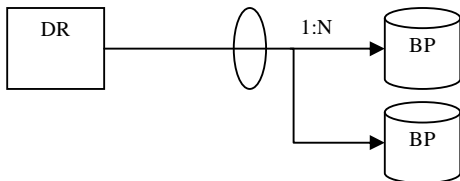
Udaljeni zahtev (*Remote Request - RR*)



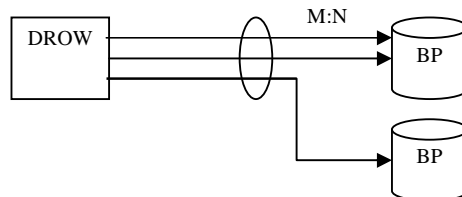
Udaljena jedinica Posla (*Remote UOW - RUOW*)



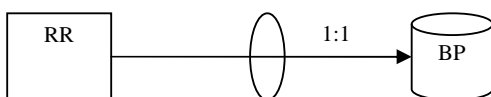
Distribuirani zahtev (*Distrib.Request - DR*)



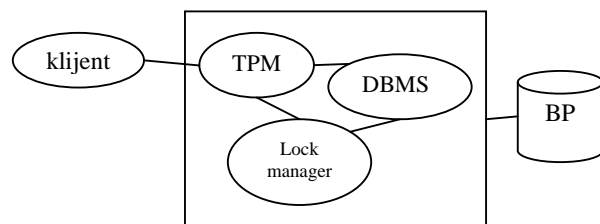
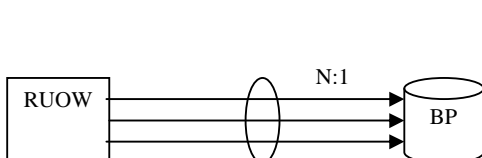
Distribuirana jedinica Posla (*Distrib.UOW - DUOW*)



58. Udaljeni zahtev

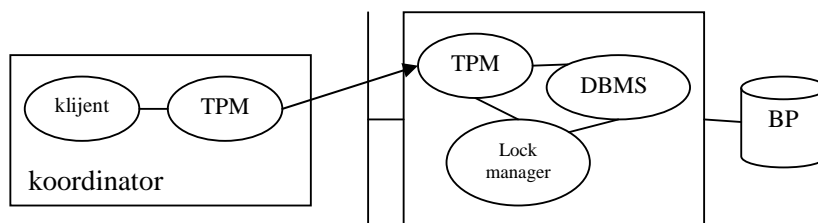


59. Udaljena transakcija



60. Distribuirani zahtev

61. Distribuirana jedinica posla

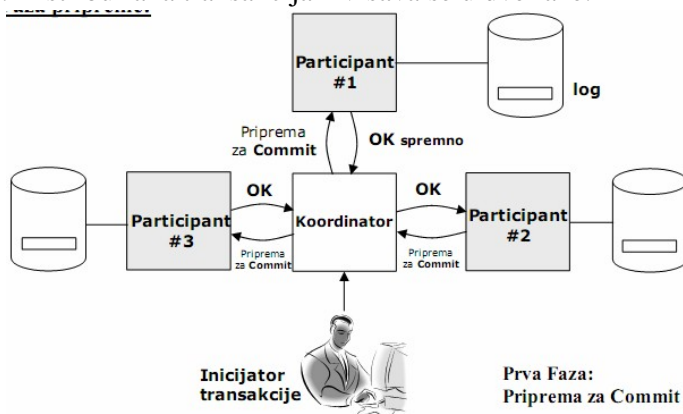


62. Protokol dvofaznog izvršavanja

2PC je protokol za izvršavanje transakcija u distribuiranom okruženju. Postoje dve vrste procesa, koordinador (inicijator transakcije) i ucesnici. Distribuirana transakcija izvršava se u dve faze:

1. Faza pripreme

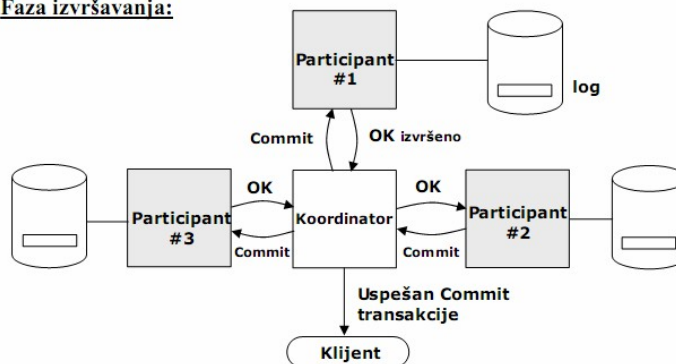
Korisnik inicira transakciju, koordinador svim ucesnicima salje poruku da se pripreme. Ukoliko svi ucesnici pozitivno odgovore ("ready") prelazi se na globalni commit u fazi 2. U suprotnom koordinador salje ucesnicima poruku "abort" i transakcija se prekida.



2. Faza izvršavanja

Kada je koordinador primio od svih ucesnika poruke, donosi odluku i upisuje je u svoj log. Tada svima salje poruku tipa "commit" ili "abort" i u svoj log upisuje poruku za kraj transakcije, a ucesnici nakon izvršenja transakcije upisuju rezultate u svoje logove.

Faza izvršavanja:



63. Osnovna svojstva distribuiranog DBMS

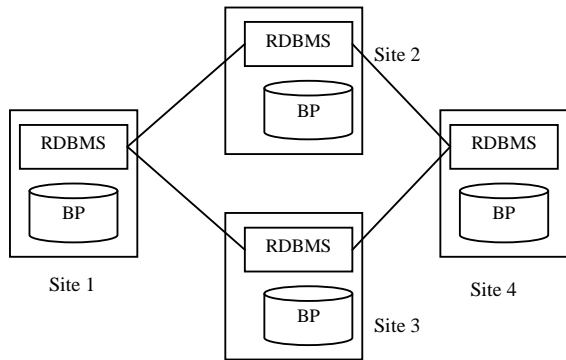
- Skup logicki povezanih deljenih podataka
- Podaci su razdvojeni u niz fragmenata
- Fragmenti mogu biti replicirani
- Fragmenti/replike mogu biti dodeljeni na lokacije
- Lokacije su povezane preko komunikacione mreze
- Podaci na svakoj lokaciji su pod kontrolom DDBMS-a
- DBMS na svakoj lokaciji moze autonomno obavljati lokalne aplikacije
- Svaki DBMS ucestvuje bar u jednoj globalnoj aplikaciji

64. Multidatabase sistemi (MDBS)

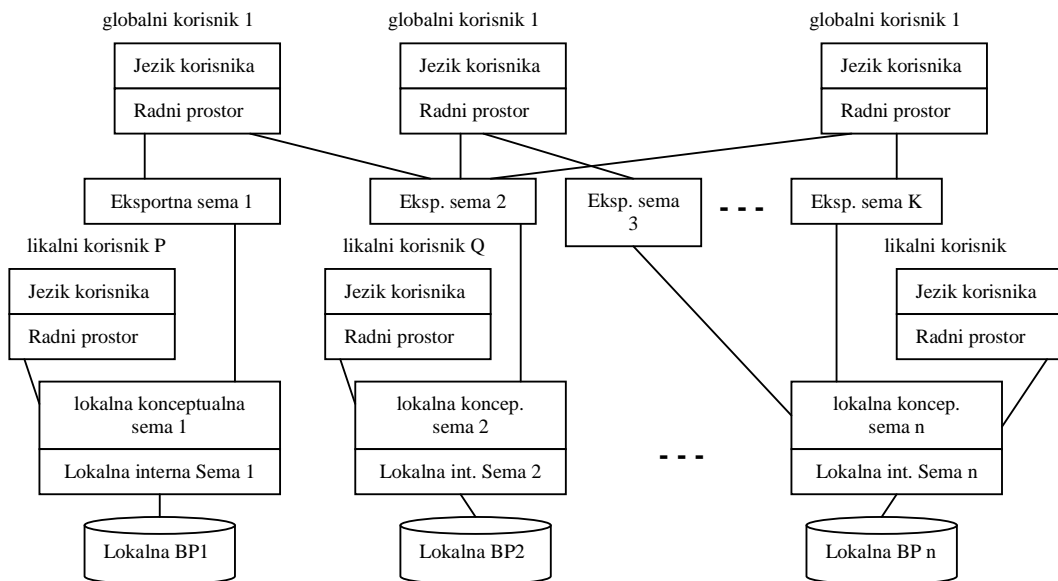
Multidatabase sistem (MDBS) je sistem sa više baza, tj. distribuirani DBMS u kome svaka lokacija održava kompletnu autonomiju. Mogu biti federalizovani i nefederalizovani.

65. Federalizovani MDBMS sistemi

- U federalizovanom database sistemu, pored globalnog postoji i lokalni korisnik.
- Federalizovani DBS predstavlja hibrid između distribuiranog (za globalnog korisnika) i centralizovanog (za lokalnog korisnika) DBMS-a.
- U slučaju federalizacije autonomnih DB servera različitih isporučilaca, koji komuniciraju po principu najmanjeg zajedničkog sadržaja, imamo dva slučaja:
 - Tesno povezani DBMS-ovi – poseduju globalnu konceptualnu semu, tj. logički opis cele BP.
 - Labavo povezani DBMS-ovi – nemaju globalnu semu, već samo eksportnu semu-pogled.



66. Arhitektura federalizovanih MDBMS sistema



67. Vlasništvo nad podacima

Pitanje 56.

68. Modeli obrade podataka

Navedene modele obrade podržavaju odgovarajuće arhitekture.

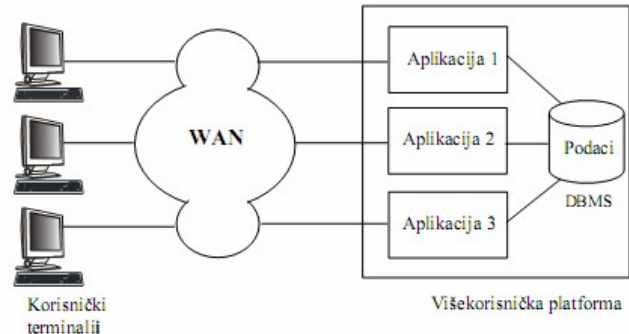
- Model centralizovane obrade - Centralizovana, više korisnicka
- Model personalne obrade -
- Model distribuirane obrade - Distribuirana, više-korisnicka

69. Arhitekture informacionih sistema

- Centralizovana, više korisnička - realizuje se putem mreže terminala priključenih na centralni (host) racunar većeg kapaciteta (MF).
- Distribuirana, jedno-korisnička – realizuje se , bilo izolovano na jednom PC racunaru bilo njihovim povezivanjem u lokalnu racunarsku mrežu.
- Distribuirana, više-korisnička - realizuje se sa više racunara, njihovim povezivanjem u lokalnu racunarsku mrežu (LAN).

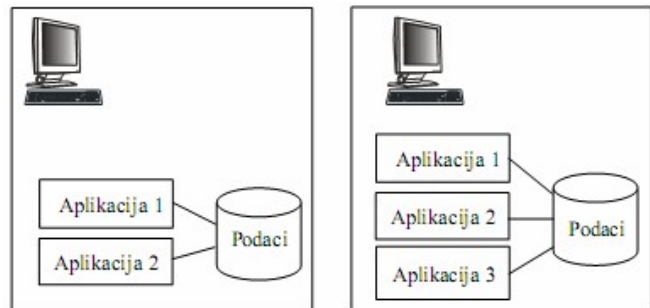
70. Centralizovana višekorisnička arhitektura

- Ova arhitektura se realizuje putem mreže terminala priključenih na centralni (host) racunar većeg kapaciteta (MF).
- Prakticno sve komponente sistema (funkcije, podaci) se nalaze i izvršavaju na centralizovanoj racunarskoj platformi i koriste od strane većeg broja (200-10000+) jednovremenih korisnika.
- Tipicno za poslovne aplikacije i IS OLTP tipa.
- Prednosti: Tehnologija stabilna, pouzdana, dobro podržana. Obezbeđuje funkcije i pristup podacima za > 1000 korisnika.
- Nedostaci: Tehnologije proizvodacke, međusobno nekompatibilne, skupe za nabavku i implementaciju, značajni troškovi dogradnje.



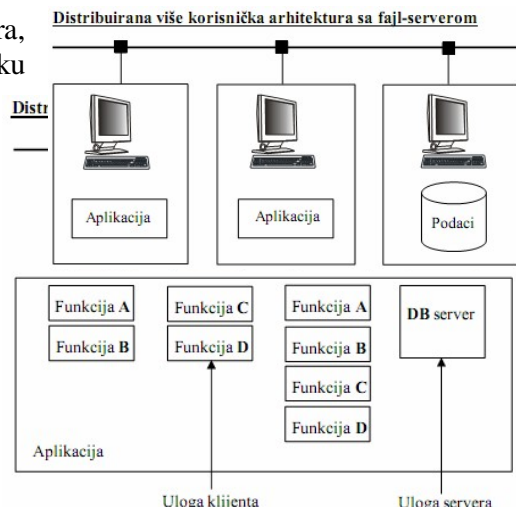
71. Distribuirana jednokorisnička arhitektura

- Ova arhitektura se realizuje, bilo izolovano na jednom PC racunaru bilo njihovim povezivanjem u lokalnu racunarsku mrežu.
- Sve komponente sistema (funkcije, podaci) se nalaze na jednom racunaru. Platformi (PC), koja je namenjena korišćenju od strane jednog korisnika.



72. Distribuirana višekorisnička arhitektura

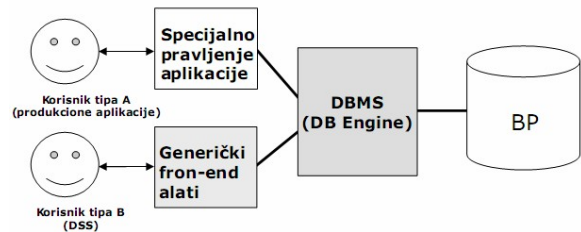
- Ova arhitektura se realizuje sa više racunara, njihovim povezivanjem u lokalnu racunarsku mrežu (LAN).
- Komponente sistema (funkcije, podaci) se mogu nalaziti na razlicitim racunarima, obicno su podaci na jednom racunaru koji ima funkciju file-servera.
- Nedostaci:



- Intezivan saobracaj kroz mrežu
- File-server postaje samo usko grlo
- Pogoršanje performansi sa porastom broja korisnika
- Prednosti:
 - Jednostavno i dobro rešenje za male info. sisteme (odeljenje, mala firma,...)

73. Tipovi (kategorije) IS, aplikacija i korisnika

- Produkcioni sistemi: OLTP tipa (On-line transaction processing)
 - Namena: pre svega zahvat/unos podat.
 - Uski skup funkcija
 - Važna pouzdanost, performanse, efikasnost
 - Režim korišćenja: 365x7x24
 - Priprema : “Front-office” (šalteri, banke, pošte, ...)
- Sistemi za podršku odlucianju: DSS tipa (Decission Support System)
 - Namena: pre svega analiza podataka
 - Širi skup funkcija
 - Cesto korišcene mat. Modela
 - Manji zahtevi za pouzdanošću
 - Fokus na fleksibilnost
 - Primena: “Back-office”
 - OLAP-On Line Analitical Processing, DW-Data Verhausing, DM-Data Mining, IIS-Intelligent IS



74. Sistemi OLTP tipa

Produkcioni sistemi: OLTP tipa (On-line transaction processing)

- Namena: pre svega zahvat/unos podataka
- Uski skup funkcija
- Važna pouzdanost, performanse, efikasnost
- Režim korišćenja: 365x7x24
- Priprema : “Front-office” (šalteri, banke, pošte, ...)

75. Sistemi DSS tipa

Sistemi za podršku odlucianju *DSS tipa (Decission Support System)*

- Namena: pre svega analiza podataka
- Širi skup funkcija
- Cesto koriste matematicke modele
- Manji zahtevi za pouzdanošću
- Fokus na fleksibilnost
- Primena: “Back-office”
- OLAP-On Line Analitical Processing, DW-Data Verhausing, DM-Data Mining, IIS-Intelligent IS

76. Prednosti i nedostaci distribuiranih IS

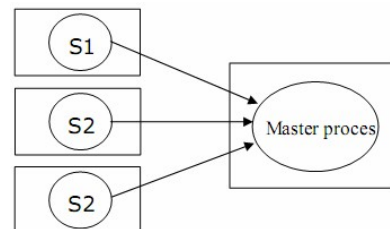
- Prednosti distribuiranih IS
 - Poboljšana fleksibilnost
 - Lokalna autonomija
 - Povećana pouzdanost i raspoloživost (kroz redundansu i fault tolerance).
 - Poboljšane performanse
 - Lokalizacija proboja sigurnosti.
 - Racunari i druge IT komponente se mogu fleksibilno locirati. Mogu se dodavati, menjati bez uticaja na druge komponente, kako bi se zadovoljile sadašnje i buduće potrebe (skalabilnost).
 - Lok. Autonomija, priznaje distribuirane prirode mnogih aktivnosti u firmi -> višestruki domeni kontrole
- Nedostaci distribuiranih IS
 - Teže ih je upravljati, zato što su znatno kompleksniji, u domenu administracije, održavanja, obezbeđenja sigurnosti.
 - Mnogo više komponenti koje potencijalno mogu otkazati. Prosečne komponente (PC) na početnu bilo znatno manje pouzdane od MF sistema.
 - Na početku, nedostatak iskusnih osoblja za podršku i razvoj, slabija podrška isporucioca. Potreba za integratorima sistema.

77. Arhitekture distribuiranih IS (DIS)

- “Master/Slave” - Master proces inicira i kontroliše svaki dijalog sa drugim (slave) procesima.
- Klijent-server - Klijent zahteva određeni servis koji obezbeđuje jedan ili više servera (procesa).
- Model od sloja do sloja (*P2P -peer-to-peer*) - Ova arhitektura eliminiše potrebu za serverima, omogućava pojedinacnim racunarima da dele resurse (aplikacije, diskove, procesor) i da međusobno komuniciraju na istom nivou.
- Grupni model (*Groupware*) - GW je softver koji podržava kreiranje, tok i pracenje nestruktuirane informacije, a kao direktna podrška kolaborativnoj grupnoj aktivnosti.
- Model distribuiranih objekata - Objekti komuniciraju putem poruka koje pozivaju neki iz skupa metoda koji definiše interfejs objekta.
- Model multimedijalnog toka - se može opisati kao kontinualni medijum sa dobro definisanim početkom i krajem koji se odigrava sa definisanom brzinom i pokazuje ne struktuirano ponašanje.

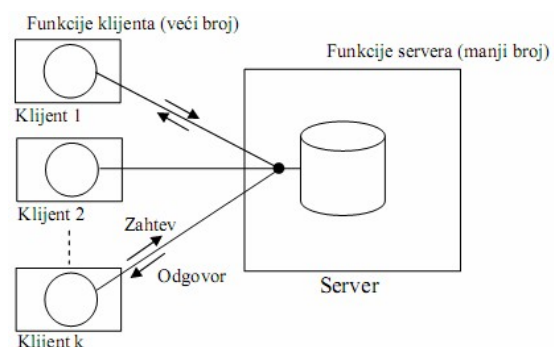
78. Master-slave model

- Master proces inicira i kontroliše svaki dijalog sa slave procesima.
- Slave procesi odgovaraju na zahteve mastera, i to samo kada su pozvani.
- Master proces definiše skup komandi i odgovara na odgovore.
- Ovo je bio model na kome su se bazirali on-line centralizovani sistemi (time-sharing IS).



79. Klijent-server model

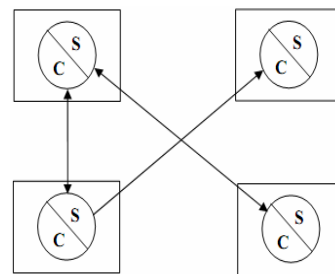
- Ovaj model je najcesce u upotrebi.
- Klijent zahteva određeni servis koji obezbeđuje jedan ili više servera (procesa).



- Serverima se pristupa preko dobro definisanog interfejsa koji mora biti poznat klijentima (server može imati veci broj interfejsa).
- Interakcija klijenta i servera se odigrava po principu zahtev/odgovor, korišćenjem odgovarajućeg protokola.
- Klijentski i serverski procesi su ravnopravni
- Manje broja servera bolja upravljivost.

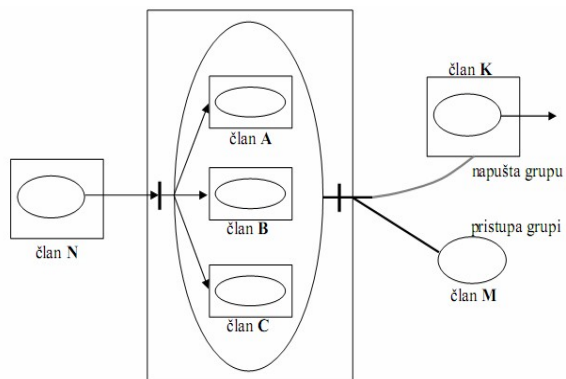
80. Model od-sloja-do-sloja (P2P)

- Ova arhitektura eliminiše potrebu za serverima, omogućava pojedinacnim racunarima da dele resurse (aplikacije, diskove, procesor) i da medusobno komuniciraju na istom nivou.
- Svaki ucesnik ima jednake mogucnosti i odgovornosti.
- Primer: deljenje fajlova i resursa (npr. Printeri), Napster.
- Nedostaci: jednostavnije i jeftinije od C/S ali otvaraju brojna pitanja u pogledu:
 - Performansi, mogu biti lošije pri velikom opterecenju
 - Sigurnosti mreže, pri odsustvu trad. administracije i zaštite



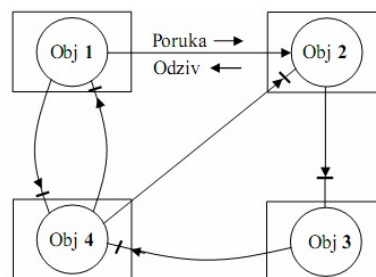
81. Grupni model (Groupware)

- GW je softver koji podržava kreiranje, tok i pracenje nestruktuirane informacije, a kao direktna podrška kolaborativnoj grupnoj aktivnosti.
- GW je ukljucen u upravljanje informacijama i aktivnostima.
- Razlike u odnosu na DBMS:
 - GW se bavi visoko nestruktuiranim podacima
 - Organizuje ih u dokumente - bazicnu jedinicu upravljanja
 - GW pomera dokumente putem e-mail i DB replika
 - Kreira baze dokumenata.
- Postoji skup tehnologija koje omogućavaju predstavljanje kompleksnih procesa koji su centrirani oko kolaborativnih ljudskih aktivnosti od kojih su 5 bazicne tehnologije:
 - Upravljanje multimedijalnim dokumentima
 - Workflow (tok posla, odvijanje posla)
 - E-mail
 - Konferencije
 - Planiranje (scheduling)
- Ni jedan GW proizvod za sada ne obuhvata sve tehnologije
- Primeri: MS Exchange, Lotus Notes/Domino 5.0, Novell GroupWise (nad Netware 5.0)
- Koristi se u slucajevima gde grupa procesa treba da saraduje na taj nacin što jedan proces treba da pošalje poruku svim ostalim procesima u grupi i dobije odgovor od 1 ili više članova npr. video konferencije.



82. Model distribuiranih objekata

- Objekti se opisuju atributima i metodama.
- Objekti komuniciraju putem poruka koje pozivaju neki iz skupa metoda koji definiše interfejs objekta. Na taj nacin se



- bez poznavanja načina implementacije objekta mogu koristiti njegovi metodi.
- Objekat može pozivati druge objekte, formirajući tako mrežu poziva objekata.
- Objekti mogu tražiti ali i pružati servise.
- Popularni standard za implementaciju distribuiranih objekata je OMG CORBA
- Prednosti:
 - objekat je prirodna jedinica distribucije
 - obezbeđuje osnovnu za integraciju DIS
 - veća produktivnost razvoja/održavanja

83. Distribuirani objekti i komponente

Distribuirani objekti su u osnovi mali aplikacioni programi koji koriste standardne interfejsa i protokole za međusobnu komunikaciju.

84. Java Beans komponentni model

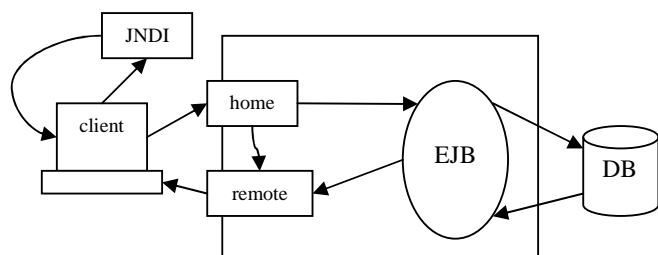
JavaBeans specificira komponentni model serverske strane. Koristeći se skupom klasa i interfejsa iz `javax.ejb` paketa, developer (starac, onaj koji razvija) može da kreira, sklopi i razvije komponente takve da se poklapaju sa EJB specifikacijama.

85. Objektni transakcioni monitor (OTM)

86. Komponente serverske strane

87. Enterprise Java Beans-i

Enterprise JavaBeans (EJB) 1.1 specifikacija definiše arhitekturu za razvoj i razvijanje transakciono distribuiranog objekta aplikacijski baziranog, s/w komponente sa serverske strane. Organizacija može da izgradi svoje lične komponente ili da kupi



komponente od nekog treceg prodavca. Ove komponente sa serverske strane, nazvane “Enterprise Beans”, su distribuirani objekti koji su domaci (host) u Enterprise JavaBeans kontejneru i obezbedjuju udaljene servise za klijente distribuirane sirom mreze.

Enterprise JavaBeans je upravljana komponenta serverske strane za modularnu konstrukciju aplikacija za preduzece.

88. Poslovni objekti

89. 3-slojna c/s arhitektura sa distribuiranim objektima

90. EJB i CORBA

92. Java platforme (J2EE)

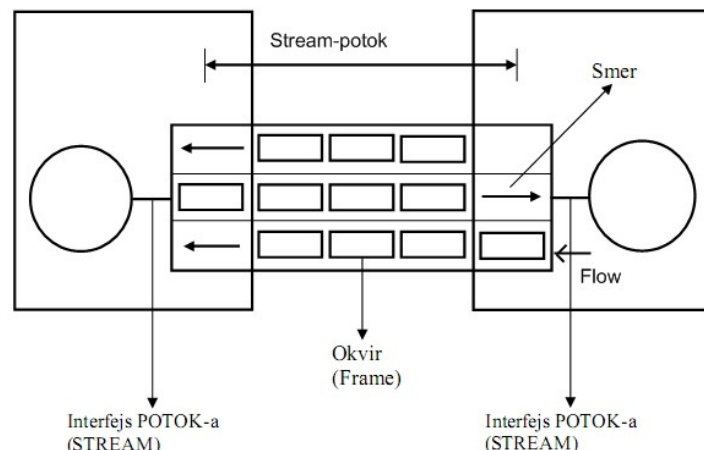
Java platforma je ime za skup povezanih programa ili platformi, od Sun-a koji dozvoljavaju razvijanje i pokretanje programa pisanih u Java programskom jeziku. Platforma nije specifično za bilo koji procesor ili operativni sistem, ali ipak prilicno na izvrsnom engine-u koji se naziva “virtualna masina” i kompajleru sa setom standardnih biblioteka koje su implementirane za razlicite hardware i oprativne sisteme, tako da se Java programi mogu startovati identicno na svima njima.

Razlicita izdanja platformi su dostupna:

- Java ME (Micro Edition): Specificira nekoliko razlicitih setova biblioteka (znane kao profili) za uredjaje koji su prilicno ograniceni da nabave/iskoruce pun set Java biblioteka. *Specifies several different sets of libraries (known as profiles) for devices which are sufficiently limited that supplying the full set of Java libraries would take up unacceptably large amounts of storage.*
- Java SE (Standard Edition): Za opstu namenu na desktop PC, serverima i slicnim uredjajima.
- Java EE (Enterprise Edition): Java SE plus raznolik API koristan za visestruki niz klijent-server aplikacija za preduzeca.

93. Model multimedijalnog toka

- MM tok se može opisati kao kontinualni medijum sa dobro definisanim pocetkom i krajem koji se odigrava sa definisanom brzinom i pokazuje ne struktuirano ponašanje.

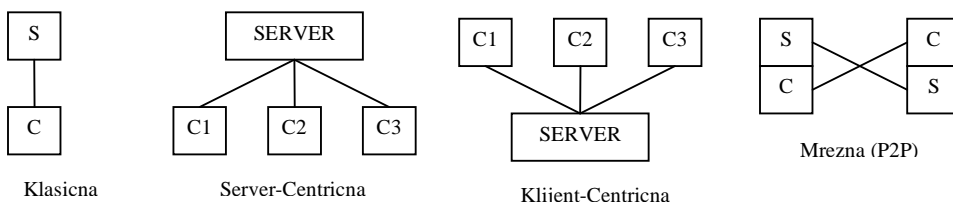


- Generalizovana arhitektura MM toka, uključuje sledeće koncepte:
 - Interfejs MM toka je samo jedan, onaj u kome su sve interakcije tokova (flow) (npr. Interfejs video konferencijskog toka se sastoji od 3 toka: audio, video, podaci)
 - Tok ima skup frame-ova
 - Frame-ovi se emituju od strane “proizvodaca”, citaju od strane “potrošača”.

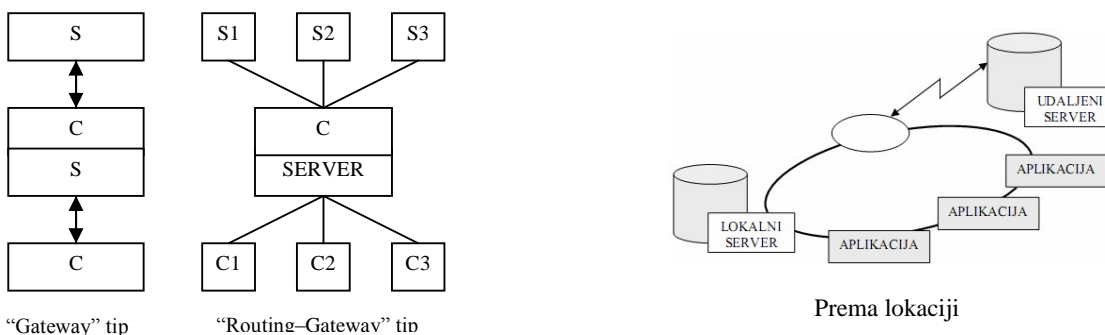
94. Klasifikacija klijent-server sistema prema hijerarhiji, lokaciji

Prema hijerarhiji:

- Dvonivojska c/s arhitektura (klasična, Server-centricna, Klijent-centricna, Mrežna (P2P))



- Višenivojska c/s arhitektura



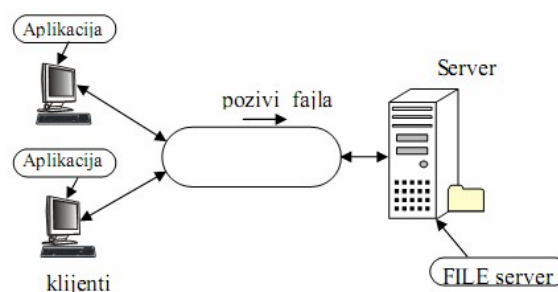
Prema lokaciji

- Lokalni server tip - Svi serveri na jednom LAN-u su lokalni serveri, njihovi tipični korisnici su Radne Grupe (workgroup) povezivane zajedničkim poslom i podacima. Lokalni Server koji opslužuje radnu grupu se naziva server radne grupe (WG server).
- Udaljeni server tip - Server udaljen u odnosu na dati LAN, koji se pojavljuje kao server za njegove klijente, se naziva udaljeni server.

95. Klasifikacija klijent-server sistema prema nameni servera

- File serveri - omogućava zajedničko korišćenje fajlova.
- DB serveri - Na DB serveru podaci, na klijentu se mora napisati apl. ili kupiti upitni alat.
- Transakcioni serveri - klijent pokreće udaljene procedure (servise) koji se nalaze na SQL (DB) serveru
- Serverske grupe – upravljanje polustrukturiranih informacija tipa: tekst, slika, mail, tok posla (workflow).
- Objektni aplikacioni serveri - klijent pokreće/poziva metode na udaljenom serverskom objektu.
- Web aplikacioni serveri - Server vraća dokumente kada ih klijent traži po imenu.

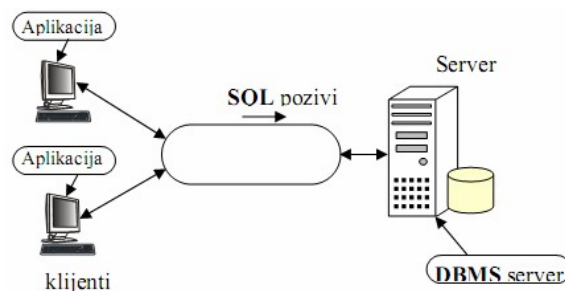
96. Fajl server



- Ova arhitektura omogućava zajedničko korišćenje fajlova.
- Intezivan saobracaj na mreži jer FS pošalje ceo fajl koji klijent lokalno potražuje.
- Arhitektura dobro za deljenje repozitorijuma dokumenata, slika, inž. crteža i drugih velikih data-objekata.

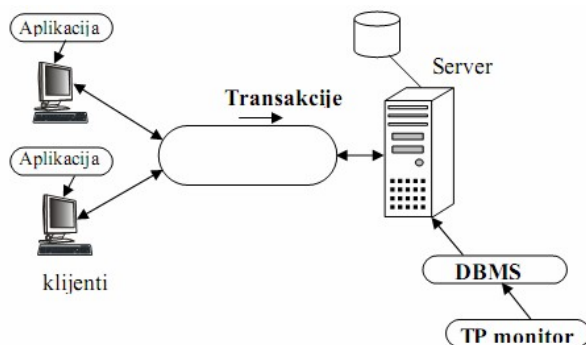
97. DB server

- Na DB serveru se nalaze sami podaci (BP) i kod za upravljanje tim podacima (DBMS).
- Na strani klijentaa se mora napisati kod za klijentske aplikacije, ili se mora kupiti upitni alat.
- Klijent salje SQL zahteve kao poruke DB serveru, a rezultati svake SQL naredbe se vraćaju preko mreže.
- Ova athitektura je osnova za DSS sisteme i DW.



98. Transakcioni server

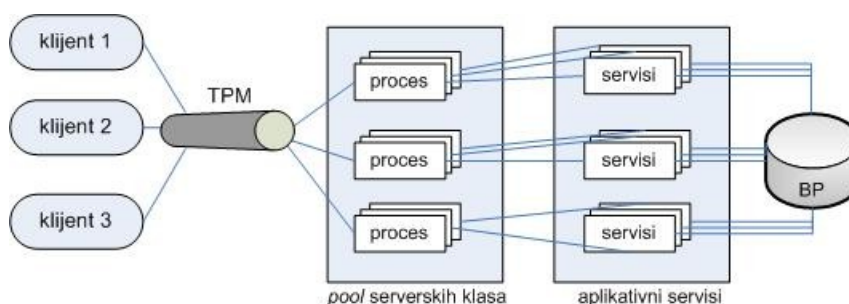
- Sa transakcionim serverom klijent pokrece udaljene procedure (servise) koji se nalaze na SQL (DB) serveru koje izvršavaju grupu SQL instrukcija.
- Saobracaj po mreži se sastoji od jedne proste poruke (zahtev/odgovor tipa), tj. SQL komande su agregirane u transakcije.
- Potrebno je napisati kod za c/s aplikacije i sa klijentske i sa serverske strane.
- Klijenti su obicno GUI tipa (Graphic User Interface), dok su na serveru aplikacije OLTP tipa.
- Postoje dve forme OLTP-a (On-Line Ttansactional Processing):
 - TP-lite, na bazi store procedura.
 - TP-heavy, na bazi TP Monitora.



99. Kanalisanje i TPM

- TPM (Transaction Processing Monitor) obezbeđuje da svi sistemi povezani sa transakcijom - učesnici, posle njenog završetka ostaju u konsistentnom stanju. TPM je tzv. „OS za obradu transakcija“, tj. obezbeđuje OS iznad OSa, koristeći *pool* deljenih serverskih procesa za brojne klijente (od 100 do 1000).
- TPM dobro upravlja:
 - procesima: pokretanje serverskih procesa, kanalisanje taskova prema njima, nadzor njihovog izvršavanja, balansiranje opterećenja,
 - transakcijama: garantuje ACID svojstva svim programima pod njegovom zaštitom,
 - C/S komunikacijom: omogućava klijentima i servisima da pozivaju aplikacione komponente na razne načine (queing, zahtev/odgovor, publish & subscribe,...)

TPM kanalisanje (*funneling*)



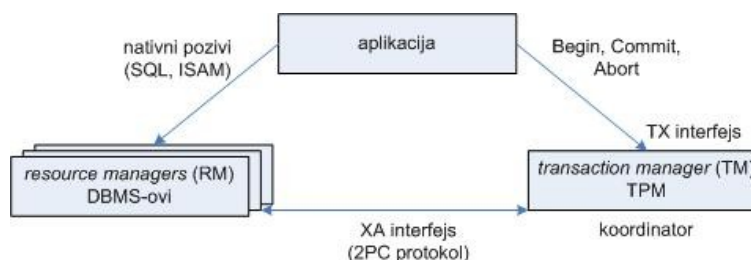
- Mnogobrojne zahteve TPM raspoređuje i “kanališe” po serverskim procesima. Određeni servisi koji su aktivni i trenutno slobodni se povezuju sa određenim (odgovarajućim) procesnim klasama. Kada klijent pošalje zahtev za servisom, TPM ga predaje raspoloživom procesu u *pool*-u serverskih klasa.

100. Standardi za TP monitore

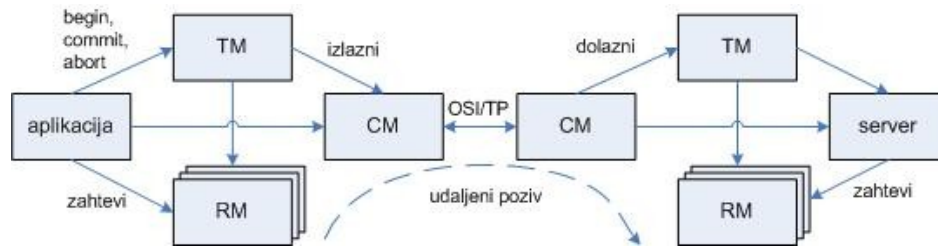
- Kako je TPM u osnovi softver za povezivanje, potrebni su standardi koji specificiraju kako se TPM povezuje sa menadžerom resursa, drugim TPMovima i svojim klijentima.
- X/OPEN DTP (*Distributed Transaction Processing*) radna grupa je specificirala programske interfejske za transakcionu obradu, koja ima tri osnovne komponente koje interaguju:
 - menadžera resursa (DBMS),
 - transakcionog menadžera i
 - aplikaciju

X/OPEN interfejsi: XA, TX

Aplikacija putem TX interfejsa komunicira sa TMom, definiše obuhvat transakcije, kao i informaciju kada da počne, komituje ili prekine transakciju. TM komunicira sa menadžerom resursa putem XA interfejsa, za koordinaciju transakcija.



U distribuiranom okruženju DTP model se mora proširiti menadžerom komunikacija – CM. Komunikacija između CM-ova odvija se po OSI/TP protokolu. OSI/TP protokoli omogućavaju heterogenim TM-ovima da rade zajedno u cilju koordinacije transakcija. XA TMI, TX RPC, CPI-C su transakcioni programski interfejsi za različite TM-ove.

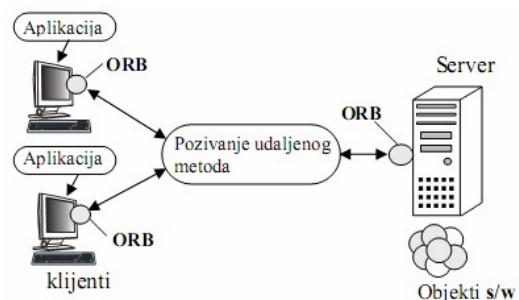


101. Komparacija TPM i memorijskih procedura

- TP heavy model baziran je na upotrebi TP Monitora. Njegove funkcije su:
 - Upravljanje procesima
 - Balansiranje opterećenja
 - Sinhronizacija globalnih transakcija
- TP light model baziran je na store procedurama i predstavlja integraciju funkcija TPM-a sa DBMSom.
- TP light je sasvim koristan u situacijama gde postoje BP jednog isporučioaca i mali do srednji broj korisnika. Idealno za početne sisteme zbog male složenosti.

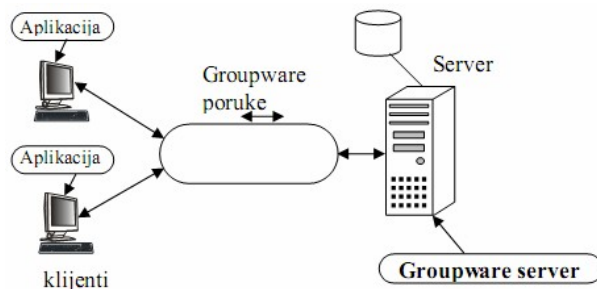
102. Objektni aplikacioni server

- Sa objektnim serverom, c/s aplikacija je napisana kao skup objekata koji komuniciraju.
- Klijentski objekti komuniciraju sa serverskim objektima korišćenjem ORB-a (Object Request Broker - Objektni Raspodeljivac), tj. klijent pokrene/poziva metode na udaljenom serverskom objektu.
- ORB locira instancu te kalse serverskog objekta, pokrene zahtevani metod, i vraća rezultat klijentskom objektu (RMI – Remote Method Invocation).



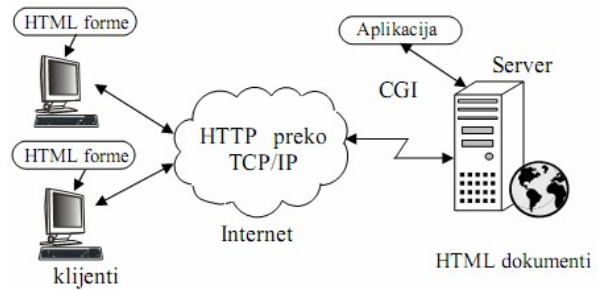
103. Grupni server

- GW adresira upravljanje polustrukturiranih informacija tipa: tekst, slika, mail, tok posla (workflow).
- C/S sistemi ovoga tipa stavljaju ljude u direktan kontakt sa drugim ljudima.
- Mnogi GW produkti danas koriste email kao srednji sloj. Cesto se Internet posmatra kao MW platforma za groupware.



104. Web aplikacioni server

- www omogućava c/s aplikacije u kojima je klijent “supertanak”(klijent samo sa browser-om), pa time i portabilan, a server “debeo”.
- Server vraća dokumente kada ih klijent traži po imenu.
- Komunikacija se obavlja putem RPC-olikog protokola (HTTP) koji definiše skup komandi, gde se parametri prosleđuju kao stringovi.
- Nova generacija: integracija web-a i distribuiranih objekata, tkz. Object web.

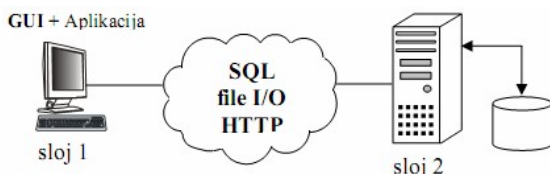
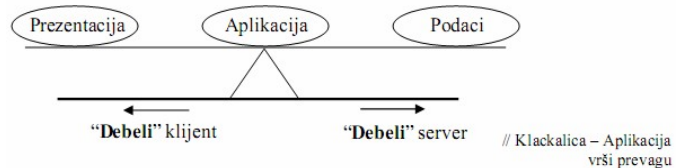


105. Komponente distribuiranih IS

- Podaci - Ova komponenta se bavi strukturama i funkcijama za cuvanje i manipulisanje podacima. Sastoji se od brojnih objekata podataka (data objects) koji reprezentuju razne tipove medija: RDB, graficki fajl, audio fajl, ili multimedia tok podatak.
- Obrada - Ova komponenta se bavi obradom objekata podataka. Može sadržati procesne objekte kao što su: Viewers, browser, search engines, aplikativno-specifna logika.
- Presentacija - Ova komponenta se bavi vizualizacijom podataka korisnika i prihvatom njegove interakcije, na dva nivoa:
 - Korisnicki interfejs - aspekti koje korisnik direktno vidi.
 - Upravljanje prezentacijom - UI manager obezbeđuje osnovne operacije koje se koriste za pravljenje i upravljanje korisničkim interfejsom od strane aplikacije, korisnika. (Servise prikaza, Upravljanje dijalogom, API). Primer je MS Windows s/w.
- Niz zajednickih servisa:
 - Povezivanje
 - Identifikacija komponenata (Naming i adressing)
 - Administracija i upravljanje
 - Sigurnost

106. Dvoslojna C/S arhitektura

- Debeli klijent
 - na klijentu su prezentacija i obrada
 - na serveru su podaci
 - Veza: RDA ili RFA
- Debeli server
 - na klijentu je prezentacija
 - na serveru su podaci i obrada
 - Veza: poziv memorisane procedure



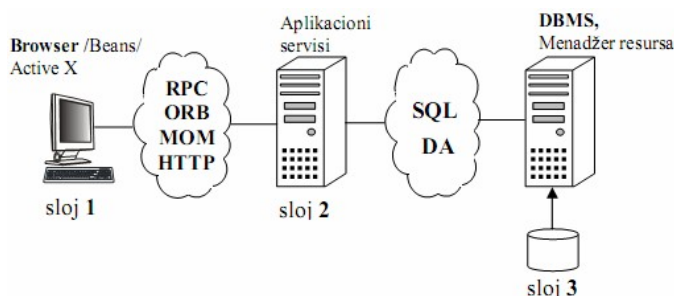
// Varijanta sa debelim klijentom.
 Za debeli server, aplikacija bi bila na strani servera, a sve ostalo bi bilo isto.

107. Troslojna C/S arhitektura

1.Sloj - Prezenciona komponenta
(veza: RDA, obr. transakc., RPC, poruke)

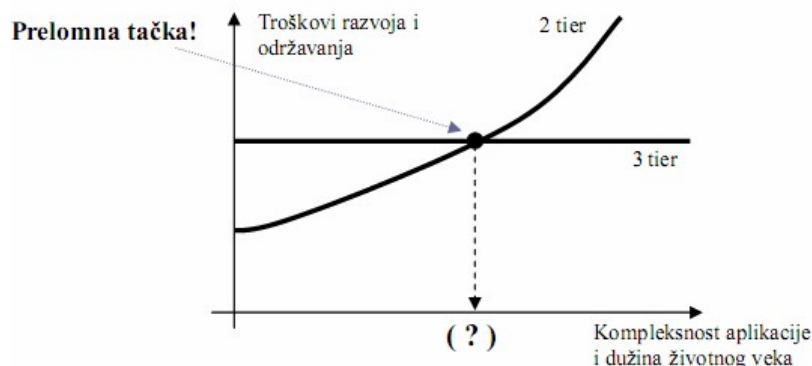
2.Sloj - Zajednicki aplikacioni servisi (AS)
(veza: RDA, obr. transakc., RPC, poruke)

3.Sloj - Zajednicki servisi podataka.



108. Komparacija 2-slojne i 3-slojne arhitekture

- 2-slojna arh. je kompleksnija, kao posledica više "logike" na klijentu kojom treba upravljati.
- Sigurnost 2-slojne je na nivou podataka, dok se kod 3-slojne može finije podešavati (na nivou servisa, metoda ili tipa objekta).
- 2-slojna arh. ima slabije performanse kao posledica intenzivnijeg SQL saobraćaja.
- 2-slojna arh. ima lošiju skalabilnost kao posledica ograničenih mogućnosti upravljanja komunikacionim linkovima klijenta.
- 2-slojna arh. nema izbor mogućnosti komunikacija, dok 3-slojna ima.
- 2-slojna arh. ima slabu podršku za internet, dok je kod 3-slojne odlična.
- 2-slojna arh. je podesna za aplikacije na nivou radne grupe/odeljenja manje groupware, dok je 3-slojna arh. potrebna za sistem od preko 300 konkurentnih korisnika.
- 3-slojna arh. koncentriše dolazeće sesije, te može distribuirati opterećenje na višestruke servere.



109. Funkcionalna dekompozicija monolitnih aplikacija

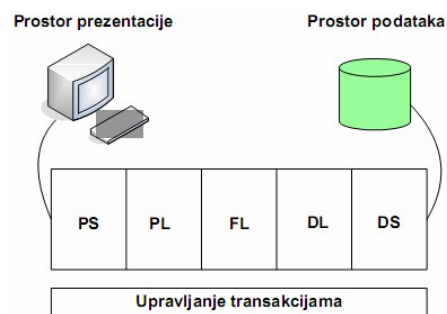
PS (prezentacioni servisi) - omogućuju prikaz informacija korisniku, određuje izgled aplikacije.

PL (prezentaciona logika) - dopunjuje PS, određuje ponašanje aplikacije. Ranije tipa DA/NE, danas obično skup procedura koje se pozivaju kao reakcija na određeni događaj (event).

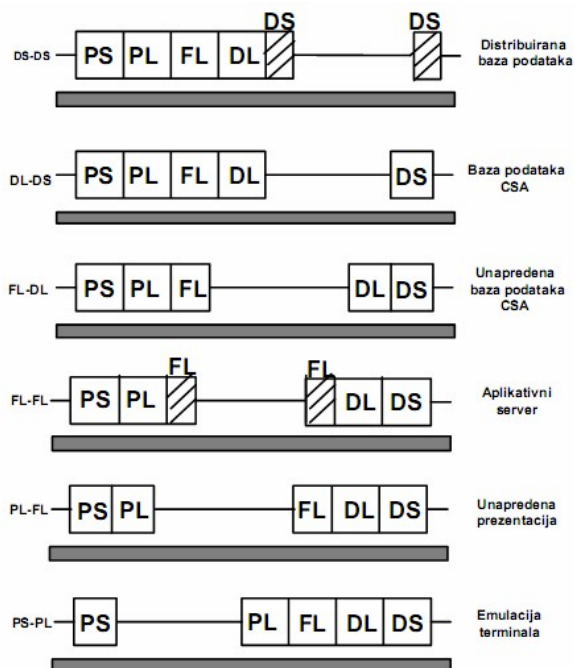
FL (funkcionalna logika) - implementira poslovna pravila aplikacije.

DL (data logika) - određuje ponašanje objekata podataka.

DS (data servisi) - upravljaju definicijom strukture i manipulacijom vrednostima objekata podataka.

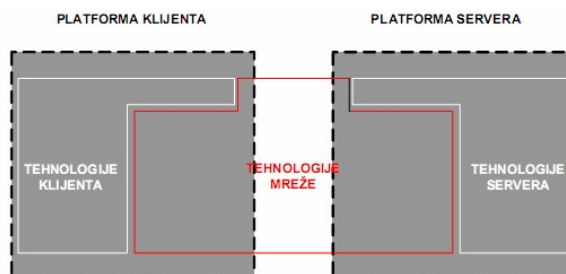


110. Moguća alokacija funkcija na elemente C/S sistema



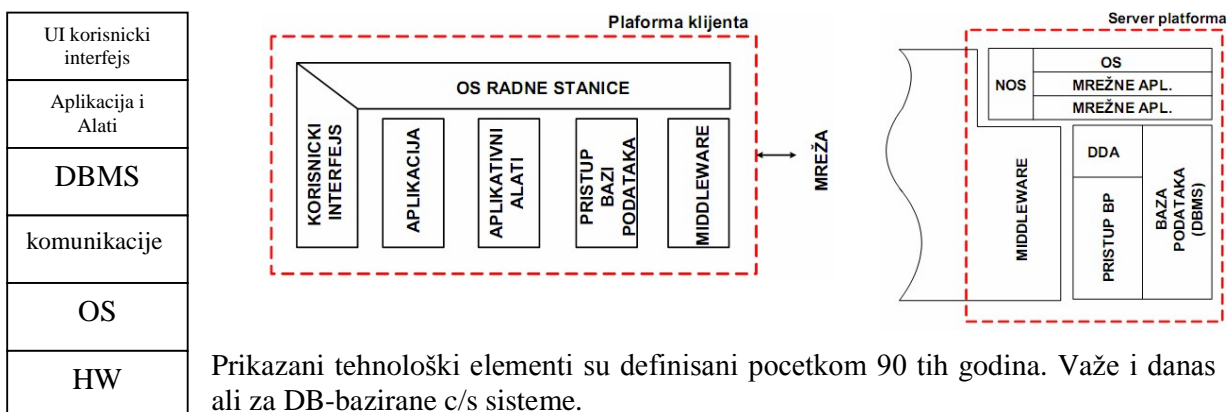
111. Osnovne C/S tehnologije

- Tehnologije klijenta
- Tehnologije servera
- Tehnologije mreža



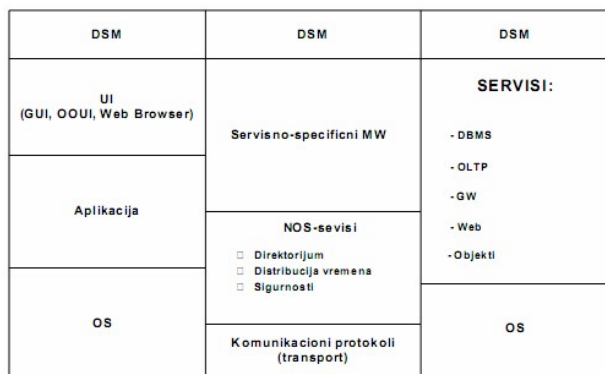
- Granice platforme se ne poklapaju sa granicama tehnologije

112. Tehnološki elementi C/S sistema



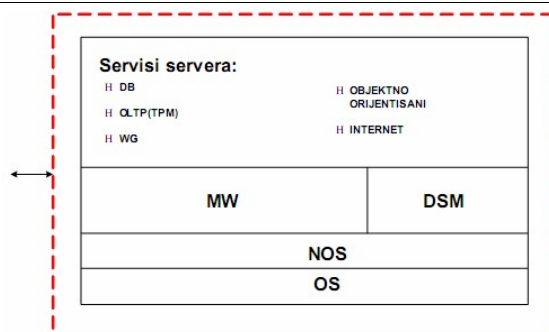
113. Generalizacija s/w arhitekture klijent-server sistema

Do novije podele doslo je sa napretkom IT i usled novih potreba (npr. DSM komponenta-Distributed System Managment), kao i zbog evolucije samih c/s sistema (od DB-servera do servera razlicitih Servisa) i razvoja middleware-a, tj. srednjeg sloja.



114. Tehnologije servera (arhitektura s/w servera, h/w platforma)

- Kod c/s sistema, funkcije zajednicke za više klijenata se obavljaju na serveru.
- Mora se raspolagati sledecim komponentama:
 - H/W platforma servera
 - Operativni sistem servera
 - Middleware (srednji sloj višeg i nižeg nivoa)
 - DSM-upravljanje distribuiranim sistemom
 - Specifnim serverskim servisima



115. OS servera

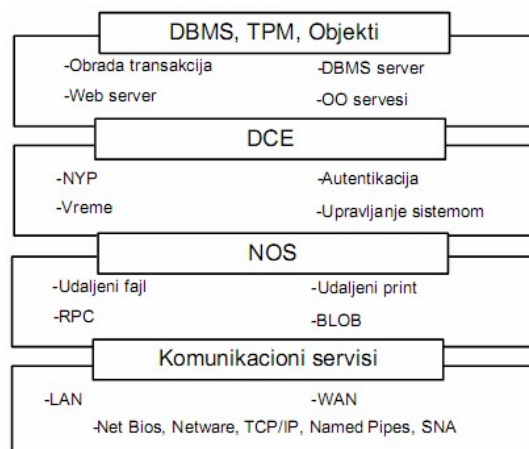
- Aplikacijama (klijentima) obezbeduje pristup resursima i upravlja periferijama.
- U okruženju distribuirane obrade, funkcije OS-a su ili bazne ili proširene. Bazni servisi su deo standardnog OS-a, dok su prošireni obicno dopunske modularne komponente.

116. Bazni servisi OS

- Preventivni multitasking – OS dodeljuje fiksne vremenske intervale za izvršavanje svakog taska i automatski obavlja prebacivanje konteksta taskova.
- Prioritet taskova – Vazno je da OS ima sposobnost tretmana taskova sa predefinisanim prioritetom.
- Semafori – Mehanizam sinhronizacije konkurentnih taskova.
- Meduprocesorska komunikacija IPC – OS mora da obezbedi mehanizam za razmenu i deljenje podataka izmedju nezavisnih procesa.
- Lokalna/udaljena IPC – OS mora da omoguci na transparentan nacin preusmeravanje IPC-ova ka udaljenom procesu. Prosirivanje IPC-a izvan granica masine je kljucno za razvoj c/s sistema (RPC).
- Niti (threads) – Niti su jedinice konkurentnosti unutar samog programa. Koriste se za kreiranje konkurentnih serverskih programa.
- Višekorisnicki fajl sistem – Mora biti takav da obezbedjuje lock-ove koji odrzavaju integritet podataka, podrzava veliki broj otvorenih fajlova, bez znacajnijeg pogorsanja performansi.
- Efikasno upravljanje memorijom – Sistem mora efikasno podrzati velike programe i velike objekte podataka, koji moraju biti lako swap-ovani sa/na disk.

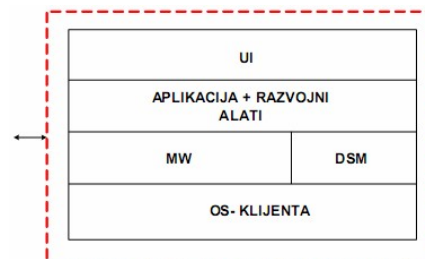
117. Prošireni servisi OS

- Različiti komunikacioni protokoli – OS mora omogućiti serveru da komunicira sa većim brojem klijentskih platformi preko različitih mreža, protokola.
- NOS ekstenzije – Treba da obezbede sredstva za korišćenje file i print servisa preko mreže.
- Binarni veliki objekti (BLOB) – tu spadaju slike, video, grafika
- Globalni direktorijumi i žute stranice (YP) – Predstavljaju ekstenzije OS-a koje omogućavaju da klijenti mogu locirati servere i njihove servise na mreži, koristeći servis globalnog direktorijuma (NDS).
- Servisi autentifikacije i autorizacije – OS mora omogućiti klijentu da dokaze serveru da je on taj za kog se predstavlja, dok autorizacija određuje da li prethodno autentifikovani korisnik ima pravo korišćenja određenog udaljenog servisa.
- Vreme u mreži – OS mora obezbediti mehanizam putem koga klijenti i serveri mogu sinhronizovati svoje interne časovnike.
- Upravljanje sistemom – Prošireni OS mora obezbediti integrisanu mrežu i platformu za upravljanje sistemom, što uključuje servise za konfigurisanje sistema, sredstva za nadzor performansi...
- Ostali servisi – DB i transakcioni, internet (http, ssl, dns)



118. Tehnologije klijenta (arhitektura sw klijenta, h/w platforma)

- Kod većine c/s aplikacija, funkcije klijenta se obavljaju na radnoj stanici krajnjeg korisnika.
- Mora se raspolagati sledećim komponentama:
 - H/W platforma klijenta
 - Operativni sistem klijenta
 - Middleware (srednji sloj višeg i nižeg nivoa)
 - DSM-upravljanje distribuiranim sistemom
 - Aplikacija+alati za razvoj
 - Korisnički interfejs
- H/W platforma klijenta predstavlja osnovnu komponentu sistema na koju se oslanjaju sve ostale. Sadrži: CPU, RAM, diskovi/CD, U/I uređaje, monitor



119. OS klijenta

- Osnovna namena OS klijenta je da aplikacijama omogući pristup h/w resursima i upravljanje interfejsima između radnih stanica i spoljnih uređaja.
- Važne karakteristike OS:
 - adresabilnost memorije
 - single vs. multitasking
 - grafički korisnički interfejs
 - h/w nezavisnost
- Poželjno je da adresabilnost bude što veća, npr. MS DOS je bio ograničen na 1 MB.

- Po svojoj prirodi, multi tasking OS su znacajno funkcionalniji i sposobniji od single tasking OS.
- Aspekti GUI-a se odnose na sposobnost OS da obezbedi aplikaciji, putem standardnog API-ja, i njenom korisniku omoguci interakciju sa samom aplikacijom i OS na intuitivan i jednostavan graficki nacin, konzistentan za sve aplikacije iz toga okruženja.
- GUI može biti ugnježđen u OS, tj. tretiran kao deo OS, npr. Mac OS, Windows NT, ili obezbeden kao proširenje OS, npr. MS Window/MS DOS, ili razlicita.
- Hardverska nezavisnost OS
 - Portabilnost OS je veoma znacajna.
 - UNIX se smatra danas najotvorenijim i najportabilnijim.

120. Korisnički interfejs klijenta

Tipovi UI klijenta:

- Ne GUI klijent – tipicno za razlicite uredjaje koji ne traze ili traze minimalnu ljudsku interakciju
- GUI klijent – sastoji se od jedne ikone i primarnog prozora sa meni barom. Ikona predstavlja aplikaciju u radu (windows 3.x, proste web strane).
- OO UI klijent – se sastoje od skupa korisnickih objekata okji kooperiraju (Windows 98, Apple Mac OS).
- Compound dokumenti – predstavljaju najnoviju OO UI tehnologiju. Svaki vizuelni objekat na ekranu je “ziva” komponenta, kojoj se moze menjati velicina, polozej, svojstva...
- Shippable place (Mesta za isporuku) – je mobilni kontejner za komponente. Mesta omogucavaju serverima da automatski azuriraju desktopove svojih klijenata (baner...).

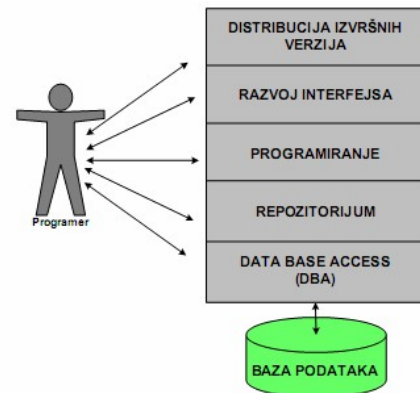
121. Alati za razvoj C/S aplikacija

- Alat je bilo kakvo razvojno okruženje, kompajler, alat za izveštavanje, ili okvir za izgradnju ili “deployment” c/s aplikacija.
- Vecina kreira aplikaciju na strani klijenta.
- U situaciji kada postoji > 200 razlicitih alata za razvoj, izbor nije jednostavan!
- Tipovi alata za razvoj, pre svega iz ugla c/s sistema DB tipa, moguca podela:
 1. 3GL alati - Tradicionalni opštenamenski programski jezici tipa: C/C++, Pascal, Cobol (MS Visual C++, Borland C++)
 2. Specijalizovani alati - IDE-Integrirana razvojna okruženja. Napravljeni za kreiranje c/s DB aplikacija. Sadrže sve potrebno za projektovanje, izgradnju i instalaciju aplikacije (Delphi, Power Builder, Visual Basic, Oracle Developer).
 3. Više-platformski alati - Cross-platform tools su slicni IDE-ovima, ali su sposobni da distribuiraju izvršne verzije aplikacije na bilo koji broj platformi (JAM 7 (JYACC Corp.), Unify, Uniface).
 4. “Smalltalk alati” - Posebna kategorija, cisti objektno orientisani koncepti. Koristi i relacione BP kao Smalltalk objekte (putem specificnog wrappera). (Object Studio (VMark Softw. inc.), Visual Age for Smalltalk (IBM)).
 5. Fajl orijentisani i desktop DB alati - zastarela kategorija. (MS Visual Fox Pro, Access, Borland Visual dBase).
 6. Alati za izveštavanje i OLAP - Tipicno SQL Report writeri omoucajavu vizuelni dizajn izveštaja i generišu potrebni SQL kod.
 7. Generatori koda - U osnovi generišu 3 GL kod koji potom ide na kompajliranje (Proto Gen).

8. CASE alati - Omogućavaju programerima da dizajniraju i instaliraju, kako aplikacije, tako i BP. Postoje CASE alati višeg nivoa (za modeliranje) i CASE alati nižeg nivoa (za generisanje koda, npr. skriptova za kreiranje tabela RDB). Primeri su Rational Rose (Rational Software), ERwin (Logic Work).
9. Alati za particioniranje aplikacije - Ovi alati omogućavaju dinamičko particioniranje sistema (aplikacije). Statiko particioniranje aplikacija-unapred fiksirano, teško promenljivo (npr. rešenja sa TPM ili rana reš. sa Distrib.obj.) Dinamičko particioniranje aplikacija- mogućnost promene alokacije delova apl. na sloj ili mašinu. (Forte (Forte SoftwareSun), Dynasty (Dynasty Technologies Inc.), Cactus (Information Builders Inc.)).
10. Web razvojni alati - Omogućavaju razvoj aplikacija koje se koriste na Internetu/Intranetu. Koriste tehnologije tipa: HTML, CGI (Common Gateway Interface), NSAPI (Netscape API), ISAPI (Internet server API), ActivX, Java,... (MS Visual Java++, Symantec Cafe Pro,...)

122. Struktura i elementi savremenog razvojnog alata

- Sloj za pristup bazi (DBA):
 - Deluje kao posrednik između ciljne BP i alata, tj. aplikacije, posle instalacije.
 - U ime alata upravlja svim DB pozivima nižeg nivoa.
 - Ovaj sloj može biti DB nezavisan i sposoban da komunicira sa brojnim BP.
- Repozitorijum-predstavlja bazu podataka o podacima, tj. bazu meta-podataka. Repozitorijum može pamti sve s/w komponente apl. uključujući kod, GUI ekrane i procese, kao i definicionu informaciju (zahteve, specifikacije, dizajn i dokument.)
- Sloj za dizajn korisničkog interfejsa je podsistem alata koji kreira ekrane, prozore i menije sa kojima korisnik interaguje.
- Sloj za programiranje je podsistem razvojnog alata koji omogućava da se prilagodi ponašanje aplikacije putem upotrebe programskog jezika.
- Sloj za deployment (instalaciju) sadrži sredstva putem kojih se aplikacija prevodi u nešto što se može izvršavati na klijentu.



123. Izbor alata za razvoj C/S aplikacija

Izbor alata uvek delikatan (>200), cene u širokom rasponu: 1.000-75.000 USD. Diskriminacija alata na bazi 5 kriterijuma, obeležija modela:

1. Poreklo alata – MF (CASE centricni), supermini racunari (4 GL alati), PC LAN (GUI centricni).
2. Tip distribuirane tehnologije - Kakve aplikacije kreira alat? Postoji veza između načina particioniranja sistema i tipa distribuiranog sistema:
 - Debeli klijent obično za DSS sisteme,
 - Debeli server obično za OLTP sisteme,
 - U sredini (npr.3-sloj) za Groupware, Distribuirane objekte ili Web.
3. Obuhvat - Koliki deo c/s funkcija obezbeđuje alat?
 - Alati specijalizovani za klijentsku ili serversku stranu, ili MW.
 - Alati za klijentsku i serversku stranu (integrisani c/s alat).
4. Model komponenti - Koji komponentni model alat podržava? Ranije, mnogi alati su imali svoj, proizvodacki komponentni model. Danas alat mora podržavati de-facto komponentne standarde. Pa tako postoje:

- Na strani klijenta: Vecina alata za debele klijente u svojoj paleti podržava ActivX. Internet klijent alati podržavaju Java Beans.
 - Na strani servera: Vecina alata podržava Enterprise Java Beans (EJB) i CORBA Beans. Neki MS COM+, Neki mešano.
5. Raspon - Koliko se dobro alat skalira za kompanijsko (enterprise) rešenje? Raspon alata je širok od alata:
- Za jedan server (za odeljenje)
 - Do multi-serverskih, ovde je posebno bitan MW na koji alat cilja (za celo preduzece).

124. "Bilderi" vizualnih mesta za 1-sloj

- Koji treba da omoguće gradnju vizualnih kontejnera (Kontejner je XML compound document obična web stranica), koji se potom mogu napuniti komponentama, kao što su ActiveX, Java Beans, Java Applet.
- Alat mora obezbediti prefabrikovane kontejnere ili mesta (vizuelna tvorevina povezanih komponenti) koja moraju biti portabilna (za razne platforme).
- Moraju biti isporuciva (shippable) preko mreže, mora imati priključke (hooks)
- Isporucivo mesto je mobilni kontejner sa komponentama, tj, mesto koje se može isporučiti preko mreže.
- Alat mora zapakovati mesta u sertifikovane JAR-ove sa ciljem distribucije.

125. "Bilderi" poslovnih objekata za 2-sloj

- Mora obezbediti serverske kontejnere sa objektnim servisima (sigurnost, persistentnost, transakcije, zaključavanje), tj. CORBA/EJB ili COM+
- Mora biti sposoban da kreira pakete na serverskoj strani, kao što su:
 - EJB JAR
 - XML instalacioni deskriptori
 - COM+ paketi - Koji sadrže komponente i njihove klase za podršku

126. Okviri (frameworks) za 3-ći sloj

- Mora obezbediti okvire koji enkapsuliraju postojeća serverska okruženja, kao što su: DBMS, GW, TPM, MOM, ERP sist., e-mail, LDAP direktorijumi i workflow.
- Ovi pozadinski servisi se obično pozivaju od strane objekata na srednjem sloju.

127. Integrirana razvojna okruženja (IDE)

Tipično je IDE namenjen jednom programskom jeziku (Visual Basic IDE), ali postoje i visejezični IDE-ovi (MS Visual Studio, Eclipse). IDE se obično sastoji od:

- Editira i izvornog koda
- Kompajlera i interpretera
- Debugger-a
- Alata za automatizaciju izgradnje
- Sistem za upravljanje verzijama
- Alati za izgradnju GUI-a
- Browser klasa
- Inspektori objekata
- Dijagram-hijerarhija klasa

128. Primeri proizvođačkih IDE-ova

- Viseplatformski proizvodjacki IDE: Code Warrior, Real basic, Multi, Maguma workbench
- Windows IDE: MS Visual Studio (podrska za C++, VB), Borland Delphi i C++ Builder, Stylus Studio XML IDE, Zeus
- Linux/UNIX IDE: Borland, Kylix za object pascal, C++
- Java bazirani IDE: Borland Jbuilder, Sun ONE Studio, Websphere Studio...

129. Primeri otvorenih (free) IDE-ova

Free s/w se odnosi na slobodu da se s/w koristi , izracunavav, adaptira, unapredjuje i javno distribuira, ukljucujuci izvorni kod, ali uz obavezu da se unapredjenje takodje ucini dostupnim. Koncept podrzava Free Software Foundation koji publikuje GNU (General Public License).

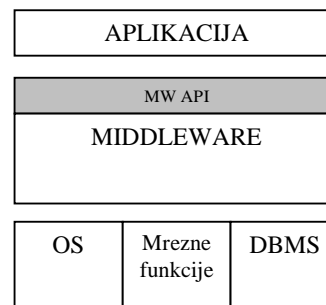
- Viseplatformski IDE: GNAT Programming Studio (Linux, Windows, Solaris, C++...), GNAVI, Emacs i XEMacs
- Windows IDE: DEV C++, Sharp Develop
- Linux/UNIX IDE: KDevelop
- Java bazirani IDE: Eclipse, JGrasp, NetBeams

130. MS .NET okvir (Framework)

- Obezbeđuje širok skup prekodiranih rešenja za tipične programske zahteve i upravlja izvršavanjem programa napisanih za ovaj okvir. Predstavlja komponentu MS Wndows OS.
- Prekodirana rešenja formiraju biblioteku klasa okvira, obuhvatajući:
 - korisnički intrfejs,
 - pristup podacima,
 - kriptografija,
 - numerički algoritmi,
 - mrežne komunikacije.
- Programi koriste funkcije iz biblioteke klasa, kombinujući ih u aplikaciju. Programi pisani za .NET okvir se izvršavaju u s/w okruženju koje upravlja *runtime* zahtevima programa, tzv. CLR (*Common Language Runtime*). CLR obezbeđuje izgled aplikacione virtuelne mašine, tako da se ne mora voditi računa o specifičnostima CPU na kome se program izvršava. Biblioteka klasa i CLR zajedno čine .NET okvir. CLR je ključni deo .NET okvira i predstavlja MS implementaciju CLI-a (*Common Language Infrastructure*).
- .NET okvir je nezavistan od platforme, korišćenjem među-jezika, slično kao i u slučaju Java VM. U osnovi je raspoloživ samo na Wndows OS.
- Tok događaja:
 - izvorni kod programa (C#, J#, VB .Net) – *just-in-time* (JIT) kompilacija,
 - među-kod (MSIL – MS Intermediate Language) – platformski-specifična (CLR) kompilacija,
 - nativni kod.
- Ciljevi dizajna:
 - interoperabilnost (obzirom da je već postojao veći broj COM biblioteka),
 - common runtime engine,
 - jezička nezavisnost,
 - biblioteke baznih klasa (BCL), tj. klase zajedničkih funkcija,
 - uprošćena instalacija,
 - sigurnost.

131. Srednji sloj (middleware) C/S sistema, namena, položaj, servisi

- MW je ključni element distribuiranih, heterogenih sistema. Predstavljaju tehnologiju, razvijenu početkom '90-ih sa ciljem obezbeđenja interoperabilnosti u heterogenom okruženju. Implementira se kao s/w koji se nalazi u sredini između aplikacije i baznog s/w sistema (OS, DBMS, mrežne funkcije).
- Upotreba zajedničkog MW API-a "maskira" kompleksnost, kako entiteta (aplikacija ili baza podataka) koji se povezuju, tako i platformi na kojima se oni nalaze. MW je u osnovi bilo koji tip s/w koji omogućava komunikaciju između dva ili više sistema.
- Cini ga skup servisa koji ima zadatak da aplikativnog programera izoluje od detalja nižeg nivoa u sferi: OS, DBMS, Komunikacionih protokola. MW obezbeđuje standardne interfejsse koji povećavaju interoperabilnost i portabilnost aplikacija.
- Tipovi MW servisa
 - Komunikacioni: omogućavaju aplik. da komuniciraju, kako međusobno, tako i sa serverima.
 - Informacioni: omogućavaju aplikacijama pristup i zajedničko korišćenje informacija u mreži.
 - Upravljački: omogućavaju pozivanje udaljenih aplik., koordinaciju izvršavanja između višestrukih apl. i upravljanje višestrukim izvršavanjem u okviru jedne apl.



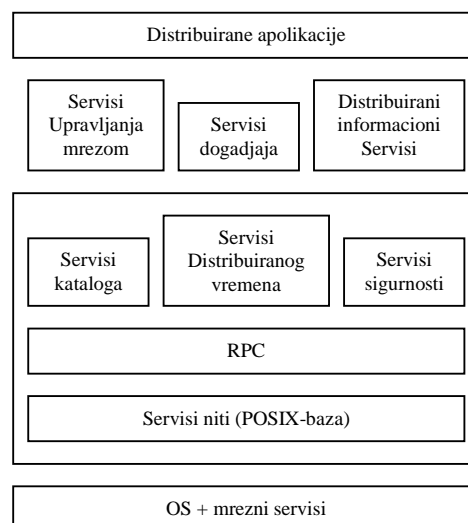
132. Celovite arhitekture srednjeg sloja, sličnosti i razlike

- Za izgradnju c/s aplikacija u heterogenim distribuiranim okruženjima:
 - OSF DCE (Open Software Foundation Distributed Computing Environment) u osnovi sadrži NOS funkcionalnost, tj. NOS na nivou celog preduzeća.
 - OMG CORBA (Object Management Group Common Object Request Broker Architecture).
- Osnovna sličnost je da su oba MW za C/S
- Osnovna razlika:
 - DCE je bio dizajniran za podršku proceduralnom programiranju, dok je
 - CORBA bila dizajnirana za podršku objektno-orijentisanom programiranju

133. OSF DCE

134. OSF DCE arhitektura

- DCE omogućava klijentu da zajednički radi sa jednim ili više serverskih procesa na drugim računarskim platformama, čak i kada su heterogene.
- DCE obezbeđuje integrisani pristup sigurnosti, imenima, i međuprocenoj komunikaciji.
- DCE obezbeđuje tehnologije:
 - Poziva udaljene procedure (RPC) - Obezbeđuje IDL jezik za def. interfejsa i kompajler.
 - Distribuirane servise imena (Naming services) - DNS omogućavaju resursima da budu identifikovani sa korisničkim orijentisanim imenima u namenskoj distribuiranoj BP koja opisuje objekte od interesa.
 - Servise vremenske sinhronizacije - Ova tehnologija obezbeđuje mehanizme za sinhronizaciju svakog računara u mreži, sa priznatim vremenskim standardom.
 - Distribuiranog fajl sistema - DCE DFS obezbeđuje uniformni prostor imena, lokacijsku transparentnost



- fajlova i visoku raspoloživost.
- Servise sigurnosti mreže (security) - DCE sigurnosni servisi obezbeđuju: autentikaciju, autorizaciju i upravljanje nalogom korisnika.
- Paket sa nitima (threads) - Tredovi su suštinska komponenta c/s sistema, koriste se od strane drugih DCE komponenti.

135. Distribuirani Naming servisi

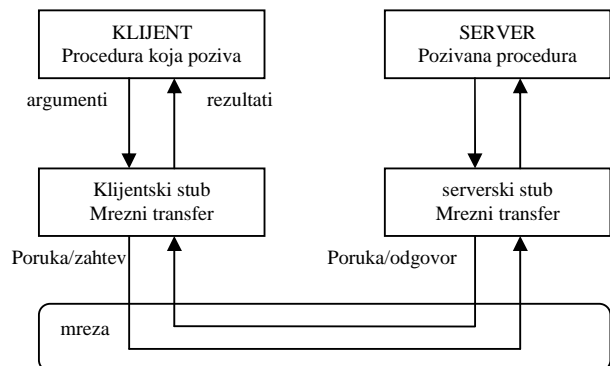
- OSF je usvojio ove servise od DEC i Siemens produkata.
- DNS omogućavaju resursima da budu identifikovani sa korisnicki orijentisanim imenima u namenskoj distribuiranoj BP koja opisuje objekte od interesa.
- Imena objekata su nezavisna od njihove lokacije u mreži.
- DCE servisi direktorijuma se sastoje od dva elementa:
 - celijskog servisa direktorijuma (CDS)
 - globalnog servisa direktorijuma (GDS)
- Obezbeđuje lokalnu naming autonomiju i globalnu interoperabilnost.
- Globalni pristup korišćenjem X.500 naming sistema ili TCP/IP Internet DNS-a (Domain Name System).

136. Tipovi middleware-a

- RPC (Remote Procedure Call) - Omogućava da se funkcija pozove iz jednog programa, a da se izvršava unutar drugog, na drugom racunaru.
- MOM (Message Oriented Middleware) - Tradicionalni MOM je tipicno quing s/w koji koristi poruke kao mehanizam za prebacivanje informacije od tacke do tacke.
- DB-orijentisan - Aplikacioni programi koriste DB MW kao mehanizam za vadenje informacija iz lokalnih ili udaljenih BP.
- Transakcioni MW (TPM, AS) - Obavljaju posao na koordinaciji transfera informacija i deljenja metoda izmedju brojnih razlicitih resursa.
- Distribuirani objekti (CORBA, COM/DCOM) - Distribuirani objekti su u osnovi mali aplikacioni programi koji koriste standardne interfejsse i protokole za medusobnu komunikaciju.
- Brokeri poruka (Message Brokers) - Omogućavaju prebacivanje informacija izmedju dva ili više resursa (izvorne ili ciljne apl.) i mogu uvažiti razlike u aplikacionoj semantici, kao i u platformama.

137. Srednji sloj RPC tipa

- Omogućava da se funkcija pozove iz jednog programa, a da se izvršava unutar drugog, na drugom racunaru.
- RPC koristi sinhroni komunikacioni mehanizam, prekida se izvršavanje programa da bi se izveo RPC "blokirajući MW" - veci "overhead" i slabije performanse.
- Klijent i server rade kao dva odvojena procesa, a procesi komuniciraju putem stub-ova.



- Stub je s/w koji obavlja konverziju lokalnih proceduralnih poziva u seriju mrežnih RPC funkcijskih poziva.
- Stub je komunikacioni interfejs koji implementira RPC protokol i specificira kako se poruke kreiraju i razmenjuju.

138. MW RPC-prezentacioni sloj

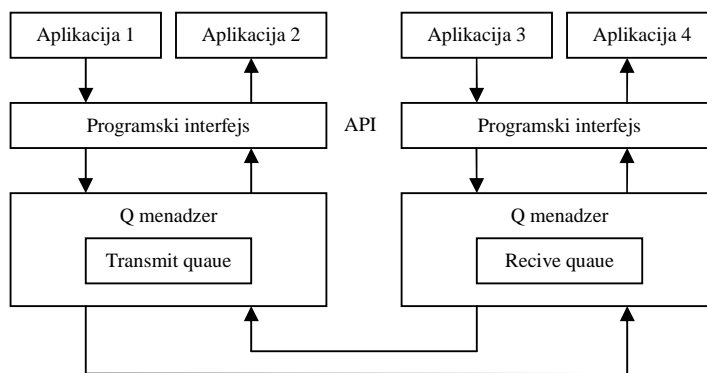
RPC prezentacioni sloj preslikava strukturu podataka aplikacije u homogenu formu koja se može prenositi. Preslikavanje podataka između aplikacije i transportne reprezentacije naziva se marshalling. Marshalling se može implementirati statički (u vreme kompilacije, stub) ili dinamički (u vreme izvršavanja).

139. MW RPC-sloj sesije

RPC sloj sesije omogućava klijentima da lociraju RPC servere. Zbog ograničenja u performansama, RPC MW se ne preporučuje za sporije mreže, tipa internet.

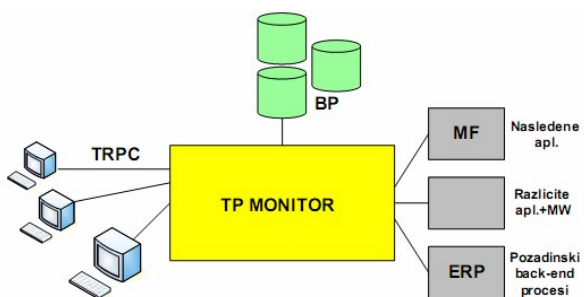
140. Srednji sloj MOM tipa

- MOM je kreiran sa ciljem prevazilaženja nedostataka RPC-a, posredstvom korišćenja poruka (message) kao mehanizma komunikacije.
- Tradicionalni MOM je tipično queing s/w koji koristi poruke kao mehanizam za prebacivanje informacije od tačke do tačke.
- Koristi se asinhroni način komunikacije koji ne blokira apl.
- Poruke su male (veličine bajtova). Moguće koristiti jedan od dva modela poruka:
 - proces-to-proces (oba proc. moraju biti aktivna)
 - queing model (zgodan za OLTP apl.)
- Varijanta MOM sa MQ (redovi poruka) omogućava da program može emitovati istu poruku mnogim udaljenim programima bez čekanja da oni budu podignuti i da rade.



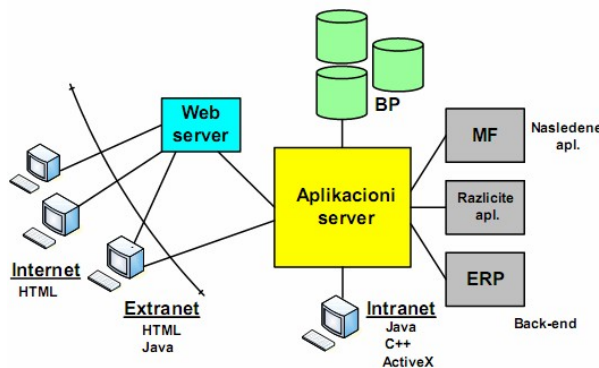
141. Transakciono orijentisan MW (transakcioni monitori)

- TP monitori su neprevaziđeni kada se zahteva podrška mnogim klijentima i visoko opterećenje transakcione obrade.
- Koriste mehanizme redova ulaznih bafera, kao zaštite od spiceva opterećenja, planiranje prioriteta, podrška serverskim nitima kao i mehanizme balansiranja opterećenja.
- TPM obezbeđuje svojstva: queing, rutiranje i slanje poruka, čime se omogućava zaobilazanje TPM-a u slučaju potrebe.



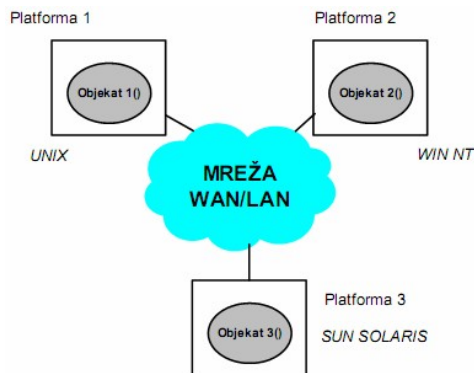
142. Transakciono orijentisan MW (aplikacioni serveri)

- Aplikacioni serveri obezbeđuju savremeni pristup u deljenju metoda kao i mehanizam za integraciju. Vecina AS se danas koristi kao Web-enabled MW, koji obradjuje transakcije od web-enabled aplikacija.
- Koriste OO jezike (Java) umesto proceduralnih koji se cesto koriste kod tradicionalnih TPM-ova.
- AS su serveri koji omogucavaju ne samo zajednicko koriscenje i obradu, nego i konekciju sa back-end resursima (npr. BP, ERP...).
- AS obezbeđuje i mehanizme za razvoj UI, kao i za spustanje aplikacije na platformu za web.



143. MW baziran na distribuiranim objektima

- Distribuirani objekti su u osnovi mali aplikacioni programi koji koriste standardne interfejsa i protokole za medusobnu komunikaciju, a smatraju se i MW pošto omogucavaju komunikaciju izmedu aplikacija, ali i tehnologiju koja podržava zajednicko korišćenje metoda na nivou preduzeca.
- Postoje dve osnovne arhitekture, kao i na njima bazirane tehnologije, alati:
 - OMG CORBA (Common Object Request Broker Architect)
 - MS COM/DCOM (Component Object Model)
- Tako jedan distribuirani objekat na UNIX serveru i drugi objekat na NT serveru mogu razmenjivati informacije ili izvršavati aplikativne funkcije putem pozivanja metoda jednog na drugom. Ovo je omoguceno time što su oba kreirana korišćenjem istog standarda (npr. CORBA) i oba koriste standardni komunikacioni protokol (npr. IIOP).
- Distribuirani objekti najbolje rade za EAI domene koji imaju potrebu da dele veliki broj zajednickih metoda.
- Najveća prednost distr. objekata je što se pridržavaju standarda za razvoj aplikacija i interoperabilnost.



144. Slojevi MW baziranog na distribuiranim objektima

- Applications
- Domain-specific MW services – vecinom proizvodjacki
- Common MW services -DO MW
- Distribution MW – OMG CORBA: ORB, EUN Java:RMI, MS-DCOM -DO MW
- Host infrastructure MW -DO MW
- OS & protocols -platforma
- h/w devices -platforma

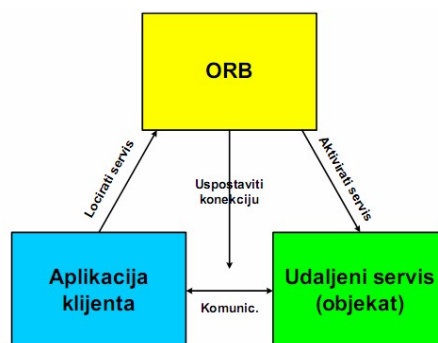
145. OMG OMA (Objet Management Architecture)

- OMA - arhitektura upravljanja objektima, sadrži opšti okvir u obliku referentnog modela.
- ORB je ključna komponenta arhitekture i omogućava objektima da interaguju u heterogenom distribuiranom okruženju.

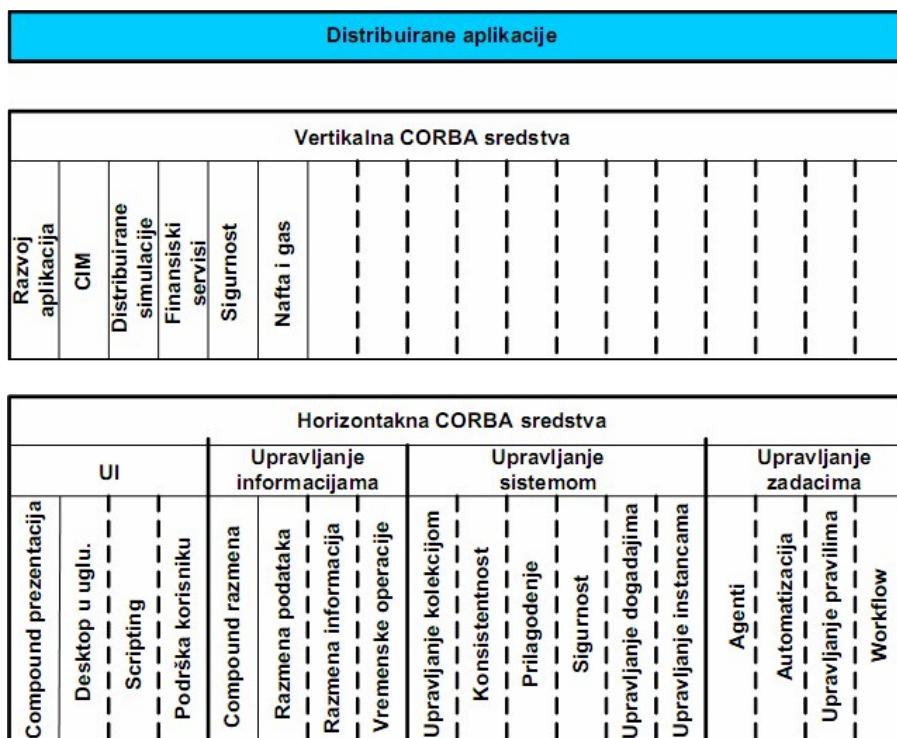


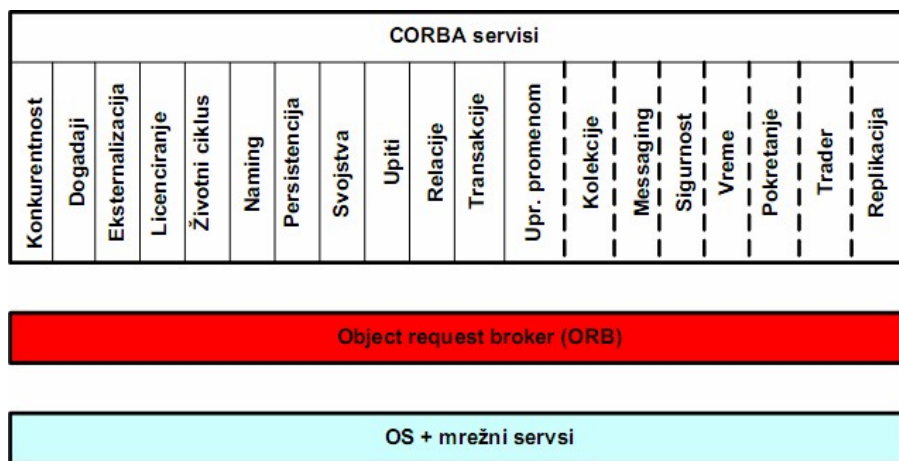
146. Middleware tipa OMG CORBA

- OMG konzorijum osnovan 1989 sa ciljem promovisanja OO tehnologije, kreira standarde i specifikacije (ali ne i s/w) koji omogućavaju interoperabilnost i portabilnost distribuiranih OO aplikacija.
- ORB je ključna komponenta arhitekture i omogućava objektima da interaguju u heterogenom distribuiranom okruženju.
- CORBA objektni bus (magistrala) definiše oblik komponenta koji žive unutar nje, kao i način kako one međusobno saradjuju.



147. Centralni deo CORBA arhitekture



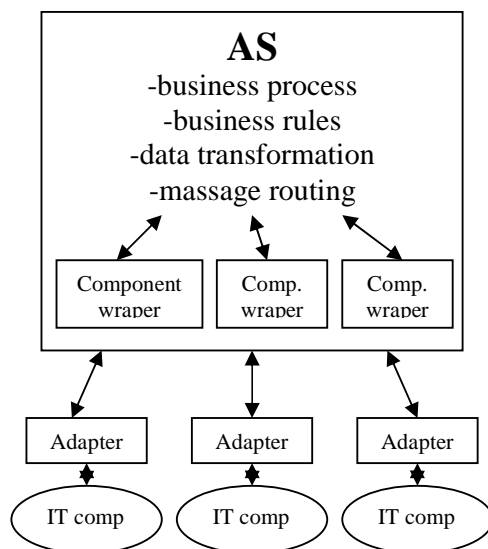


148. Povezivanje/integracija aplikacija u preduzeću (EAI)

- Motiv je prelazak sa monolitnih specijalno pravljenih aplikacija na komercijalno raspoložive aplikacije na heterogenim platformama, ili slučaj kada firma kupi manju firmu pa treba da se poveže sa njenim IS.
- Tipovi EAI:
 - Data level EAI – Omogućava pomeranje podataka između BP u situacijama kada aplikacije moraju deliti podatke iz različitih BP. Prednost je da se same aplikacije ne moraju menjati.
 - Message level EAI – Bavi se razmenom poruka između višestrukih aplikacija.
 - Process level EAI – Bavi se izgradnjom opšte-kompanijskih poslovnih procesa i uključivanjem postojećih aplikacija u te procese. Stvarna razmena se i dalje vrši putem message-inga, dok se EIA MW ponasa kao workflow masina.

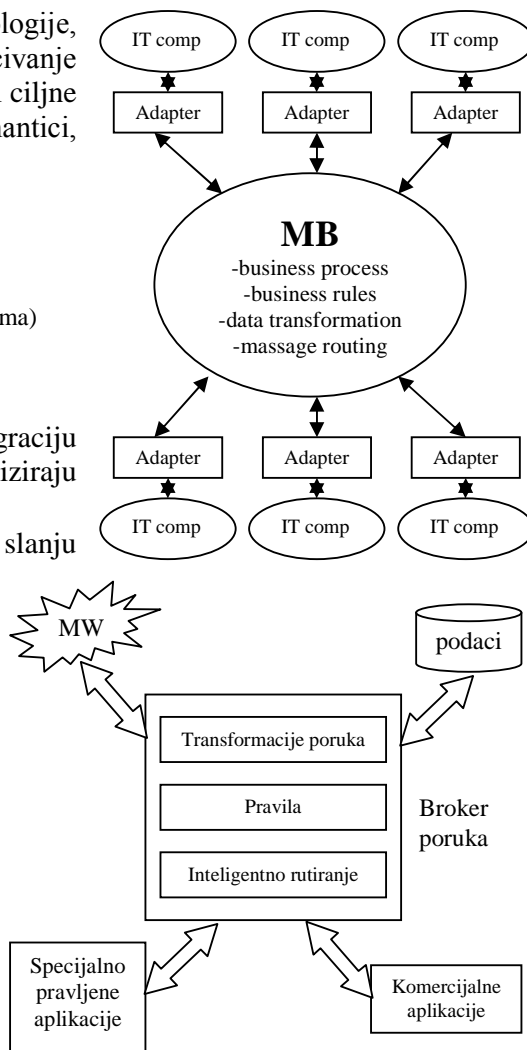
149. EAI na nivou procesa

Process level EAI se može realizovati putem AS ili message broker MB.



150. Brokeri poruka

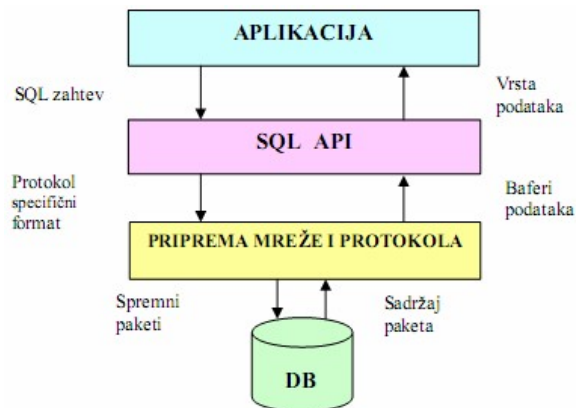
- MB se grade "iznad" postojeće MW tehnologije, žargon "midlver midlvera". Omogućavaju prebacivanje informacija između dva ili više resursa (izvorne ili ciljne apl.) i mogu uvažiti razlike u aplikacionoj semantici, kao i u platformama.
- MB obezbeđuju sledeće servise:
 - Transformaciju poruka
 - Obradu pravila (za obradu i distribuciju poruka)
 - Inteligentno rutiranje/Upravljanje tokom
 - Repozitorijuma (BP sa inf o izvornim/ciljnim aplikacijama)
 - "Warehousing" poruka (BP poruka)
 - Direktorijuma, upravljanja, APIa, Adaptera,
 - GUI (kreiranje pravila, def. transform. logike)
- MB predstavljaju (potencijalno) nirvanu za integraciju apl na nivou celog preduzeća (EAI) jer minimiziraju izmene na postojećem sistemu.
- Bazirani su na asinhronom, neblokirajućem slanju poruka.
- MB obezbeđuju servise aplikacijama putem API-a i adaptera.



- MB imaju tri osnovne komponente:
 - Transformacije poruka
 - Mašina pravila (Rules engine)
 - Inteligentno rutiranje

151. Srednji sloj za povezivanje sa BP (namena, položaj i funkcije)

- U osnovi to je MW orijentisan na BP, koji povezuje klijenta sa severom BP. Aplikacioni programi koriste DB MW kao mehanizam za unos/iznos podataka iz lokalnih ili udaljenih BP.
- U ranim fazama razvoja c/s sistema sa DB serverom, navedeni MW se nazivao DBA (Data Base Access) sloj.
- Funkcije DB MW su:
 - Aplikacioni interfejs predstavlja interfejs prema aplikaciji.
 - Konverzija zahteva, jezik apl. se prevodi u nešto razumljivo (SQL) od strane ciljne BP.
 - Slanje zahteva, sposobnost da se preko mreže pošalje upit prema BP.
 - Obrada zahteva, sposobnost da se inicira obrada upita na ciljnoj bazi.



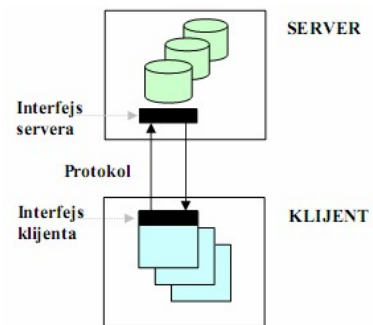
- Prebacivanje rezultata, sposobnost da se rezultatni skup (rezultat upita) vrati nazad do aplikacije.
- Konverzija rezultata, sposobnost da se rezultatni skup konvertuje u format razumljiv od strane aplikacije, klijenta, koja je postavila upit.

152. Nacini povezivanja klijenta i servera

- Nativni (proprietary) DB MW - najbolje performanse i pristup specifičnim (nativnim) svojstvima
- DB MW tipa zajedničkog interfejsa - Standardizacija zajedničkog SQL API-
- DB MW tipa zajedničkog "gateway"-a - Pored zajedničkog SQL API-ja, na klijentskoj strani postoji i zajednički gateway drajver koji u sebi sadrži konverzioni s/w.
- DB MW tipa zajedničkog protokola

153. Native/proizvodjaci DB MW

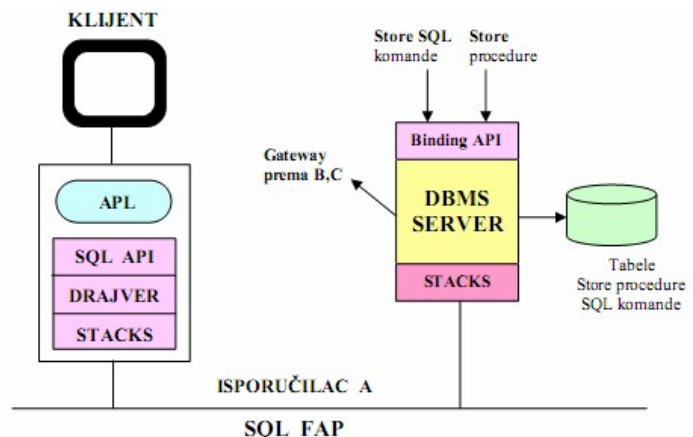
- Kreiran je za određeni tip BP (DBMS) (npr. Oracle, Sybase, DB/2, Informix) i obezbeđuje najbolje performanse i pristup specifičnim (nativnim) svojstvima (npr. store procedure, triggeri).
- Tipično su to API-i koje obezbeđuje proizvođač RDBMS-a, u obliku C i/ili C++ biblioteka.
- Nativni DB MW najbolje funkcioniše u homogenom okruženju.
- Korisceno DB MW rešenje ima više slojeva:
 - Interfejs klijenta
 - Protokol
 - Interfejs servera
 - Jedinstveni protokol za povezivanje je SQL FAP (Format and Protocol).
- Konceptija povezivanja c/s sa DB MW u homogenom okruženju
 - Interfejs klijenta, prihvata zahteve aplikacije i generiše poruku u sklopu protokola, koja se prosleđuje serveru.
 - Interfejs servera, ima ulogu da prihvati zahteve klijenta i prosleđi ih DB engine



154. Arhitektura c/s sistema sa nativnim DB MW u homogenom okruženju

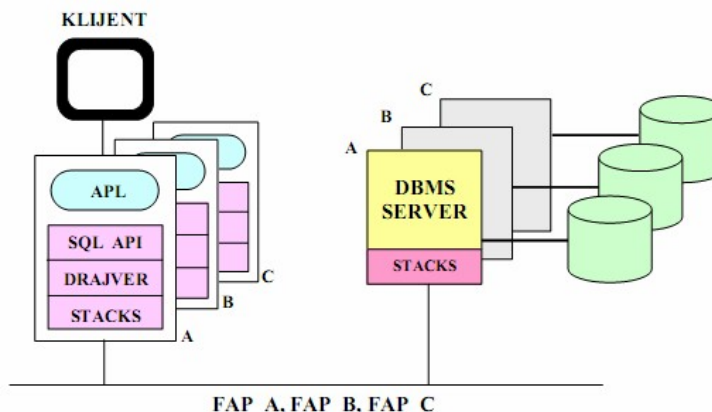
Tipično MW rešenje za homogene uslove danas obezbeđuje:

- SQL API, specifičan za datog isporučioća, koji radi na različitim platformama (h/w, OS). SQL API, se obično javlja u dve varijante:
 - ESQL ugnjezdjeni SQL API: dobre performanse, slaba fleksibilnost, za ograničeni broj programskih jezika
 - CLI-SQL API na nivou poziva: slabije performanse, dobra fleksibilnost, ne zahteva poznavanje ciljnih BP.
- SQL drajver, za implementaciju protokola koji je razvio proizvođač za komunikaciju sa njegovim DBS-om. To je tipično run-time element za klijenta, koji prihvata API pozive, formatira SQL poruku i "rukuje" razmenom podataka sa DB serverom.
- FAP podršku za veći broj različitih protokol stekova (IPX/SPX, TCP/IP,...)



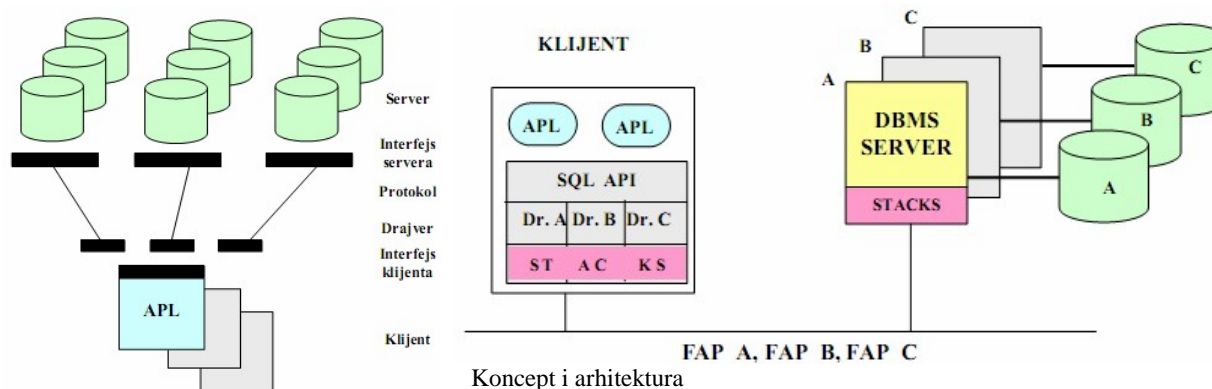
155. Arhitektura c/s sistema sa nativnim DB MW u heterogenom okruženju

- Višestruki DB drajveri troše memorijski prostor na klijentu. Pri tom se postavlja pitanje kako pratiti koje verzije drajvera su na kom klijentu?
- Višestruki FAP-ovi, bez interoperabilnosti, impliciraju da različiti DB protokoli samo fizički dele mrežu, ali ne mogu međusobno saradivati.
- Višestruki alati za administraciju impliciraju da DBA mora biti upoznat sa većim brojem radnih stanica, posebnim UI i semantikom.
- Najzad, zbog brojnih problema u heterogenom okruženju, pristupilo se standardizaciji ključnih elemenata: interfejsa, protokola, drajvera.



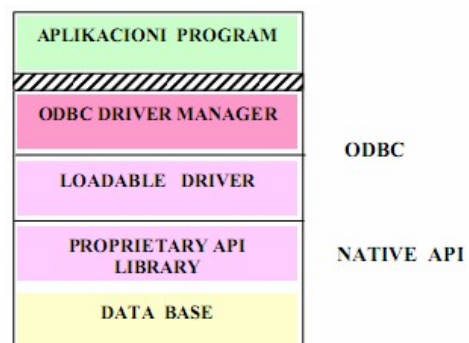
156. Implementacije tipa zajednickog interfejsa

- Standardizacija zajednickog SQL API-ja koji će koristiti sve aplikacije, a razlike u serverskoj strani će tretirati drajveri za različite BP.
- Problemi ovog rešenja su sto i dalje trebaju višestruki drajveri i višestruki FAP-ovi, apostavlja se i pitanje koji SQL API uzeti za standard?



157. Povezivanje sa ODBC

- MS Open Data Base Connectivity (ODBC) Microsoft-ova implementacija DB MW tipa zajednickog interfejsa, sa CLI-API-jem. Verzije ODBC-a:
 - MS ODBC 1.0 SDK: 1992, spor, bagovit.
 - MS ODBC 2.0 SDK: 1994, za 32 bitne racunare.
 - MS ODBC 3.0 SDK: 1996, Unicode podrška
 - MS ODBC 3.5 SDK: 1998.
- DB MW tipa ODBC-nedostaci
 - MS potpuno kontroliše ODBC standard



- Stalno se menja
- Pitanje kada ce biti uskladen sa SQL-3
- Obicno sporiji od nativnog API

158. Povezivanje sa JDBC

- JDBC je SQL API za aplikacije, applete i servlete pisane na Java-i (OO-API)
- JDBC API definise Java klase da predstavi: konekcije, SQL iskaze, rezultujuce skupove, meta-podatke BP i druge DB objekte.
- Koriscenjem JDBC, Java aplikacije i apleti mogu izvorsavati SQL iskaze iz BP.
- Postoje JDBC drajveri razlicitog tipa.

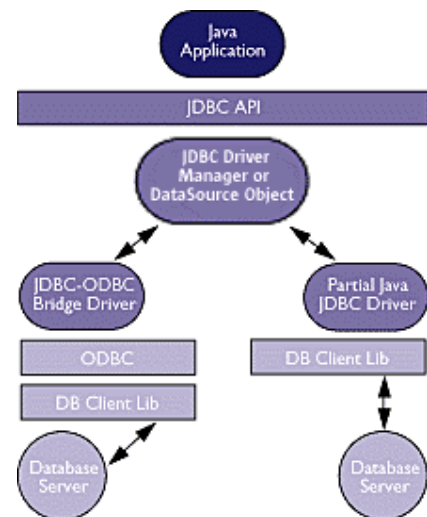
159. JDBC, osnovni delovi

- JDBC je SQL API za aplikacije, applete i servlete pisane na Java-i (OO-API)
- JDBC API definise Java klase da predstavi: konekcije, SQL iskaze, rezultujuce skupove, meta-podatke BP i druge DB objekte.
- JDBC API moze da ucini tri stvari:
 - Ostvariti konekciju sa DB ili pristupiti bilo kojem tabelarnom izvoru podataka.
 - Slati SQL iskaze
 - Obraditi rezultate
- JDBC API sadrzi dve glavna seta interfejsa. Prvi je JDBC API za one koji pisu aplikacije, i drugi je JDBC API drajver niskog nivoa za one koji pisu drajver.

160. JDBC tip 1 i 2, arhitektura

Left side, Type 1: JDBC-ODBC Bridge plus ODBC Driver - This combination provides JDBC access via ODBC drivers. ODBC binary code -- and in many cases, database client code -- must be loaded on each client machine that uses a JDBC-ODBC Bridge. Sun provides a JDBC-ODBC Bridge driver, which is appropriate for experimental use and for situations in which no other driver is available.

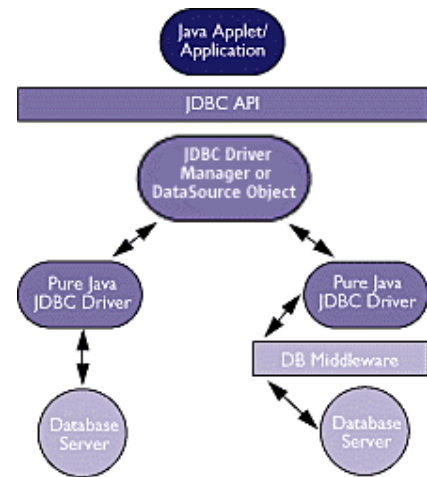
Right side, Type 2: A native API partly Java technology-enabled driver - This type of driver converts JDBC calls into calls on the client API for Oracle, Sybase, Informix, DB2, or other DBMS. Note that, like the bridge driver, this style of driver requires that some binary code be loaded on each client machine.



161. JDBC tip 3 i 4, arhitektura

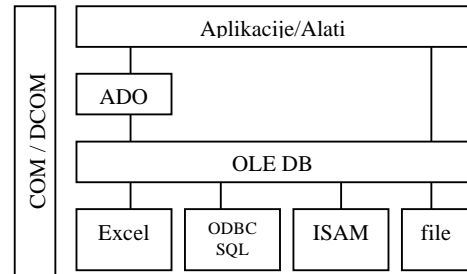
Left side, Type 4: Direct-to-Database Pure Java Driver - This style of driver converts JDBC calls into the network protocol used directly by DBMSs, allowing a direct call from the client machine to the DBMS server and providing a practical solution for intranet access.

Right side, Type 3: Pure Java Driver for Database Middleware - This style of driver translates JDBC calls into the middleware vendor's protocol, which is then translated to a DBMS protocol by a middleware server. The middleware provides connectivity to many different databases.



162. Povezivanje sa OLE-DB i ADO

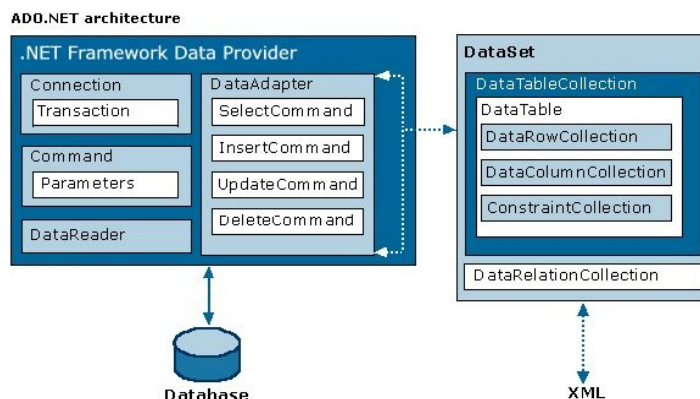
- OLE DB je rešenje za pristup podacima za Windows C i C++ programe. Obezbedjuje pristup razlicitim izvorima podataka.
- ActivX Data Objekti (ADO) je objektni sloj preko OLE DB koji obezbedjuje pristup podacima za jezike bez pointera (Visual Basic) i skript jezike tipa Java Script i VB Script.
- MS arhitektura:
 - OLE DB predstavlja interfejs podataka nižeg nivoa, obezbedjuje pristup proizvoljnom broju data resursa, kao standardnim COM objektima.
 - ADO predstavlja interfejs podataka višeg nivoa i obezbedjuje pristup podacima za jezike i bez pointera i skript jezicima.



163. MS ADO.NET

- ADO.NET je Microsoft-ova tehnologija za pristup podacima, koja omogućava aplikacijama da se konektuju na BP i koriste njene podatke. Bazirana je na .NET FRAMEWORK-u.
- ADO.NET API je dizajniran u cilju upotrebe od strane jezika koji se koristi u .NET FRAMEWORK-u, kao sto su: Visual Basic, C#, J# i Visual C++.

164. Arhitektura ADO.NET



165. Komponente MS ADO.NET aplikacije

ADO.NET sadrži dve glavne komponente:

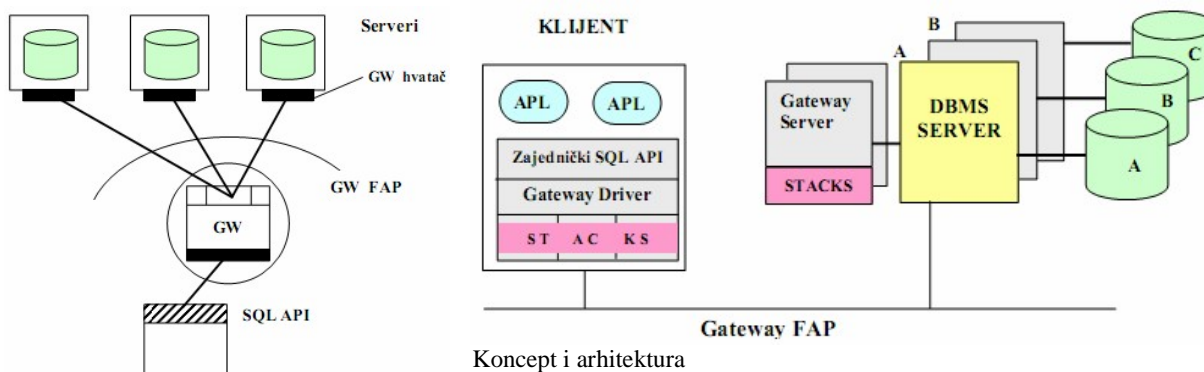
- .NET FRAMEWORK data provider – su komponente eksplicitno dizajnirane za manipulaciju podacima i brzu, read-only konekciju. Objekat Connection obezbeđuje vezu sa izvorom podataka. Objekat Command omogućuje vraćanje rezultata komandi, pokretanje i prosledjivanje parametara. Data adapter predstavlja most između Data Set objekata i izvora podataka. On koristi objekat Command za izvršavanje SQL naredbi nad BP, istovremeno “puneci” Data Set podacima.
- Data set – je dizajniran za potpuno otvoren pristup podacima svih vrsta izvora podataka. Može se koristiti sa višestrukim i heterogenim izvorima podataka, sa XML podacima ili za upravljanje lokalnim podacima u aplikaciji. Podaci se kesiraju u Data Set-u, te aplikacija može lokalno manipulirati njima.

166. Položaj i odnos ADO i ADO.NET

ADO.NET is the data access component of Microsoft's new .NET Framework. Microsoft bills ADO.NET as “an evolutionary improvement” over previous versions of ADO, a claim that has been hotly debated since its announcement. It is certainly true that the ADO.NET object model bears very little relationship to earlier versions of ADO.

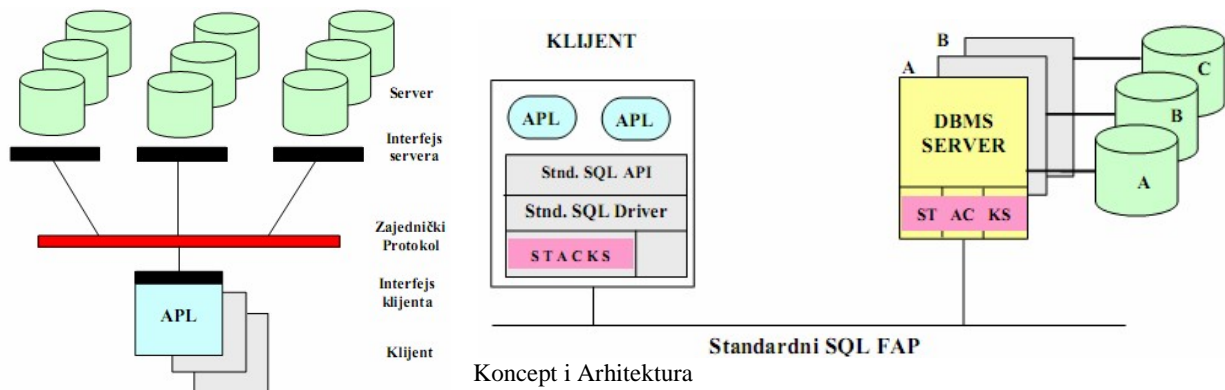
167. Povezivanje putem zajednickog gateway-a

Pored zajednickog SQL API-ja, na klijentskoj strani postoji i zajednicki gateway drajver koji u sebi sadrži konverzioni s/w. Takodje na strani servera se pravi gateway ali iz drugog razloga: npr. Oracle pravi gateway prema drugim velikim proizvođačima kako bi što više ljudi moglo brže i lakše da predje na njihova rešenja.



168. Povezivanje putem zajednickog protokola

- Najidealnija varijanta koja podrazumeva zajednicki SQL API, standardni SQL drajver i standardni zajednicki SQL FAP.
- Eliminisu se gateway “hvatači”, što pozitivno utice na performanse, cenu, održavanje.
- Zajednicki FAP mora podržavati ili super-set svih SQL dijalekata ili DB isporucioci moraju zameniti svoje FAP-ove standardnim. Iz ovog drugog razloga proistice cinjenica da ovakav protokol postoji: ISO RDA (Remote Data Access), ali ga niko ne koristi jer niko nije uskladjan sa njim.



169. Otvoreni sistemi i IT standardi, standardizacije organizacije

- Otvoreni sistem je sistem koji implementira dovoljno otvorenih (standardizovanih) specifikacija za interfejs, servise i formate podrške. Osobine otvorenih sistema:
 - Portabilnost – sposobnost sistema da radi na razlicitim racunarskim platformama (h/w + OS), zavisi od interfejsa.
 - Skalabilnost – sposobnost sistema da omoguci vise performanse kada porastu zahtevi, a putem dodavanja dopunske racunske opreme.
 - Interoperabilnost – sposobnost sistema da rade zajedno.
 - Interkonektivnost
 - Stabilnost
- IT standardi mogu biti de-jure i de-facto.
- Organizacije: Medjunarodne (ISO, ITU), Nacionalne (ANSI, FIPS), Konzorcijum (X/OPEN, OSF, OMG, W3C...)

ISPITNA PITANJA IZ PREDMETA KLIJENT-SERVER SISTEMI: 2006.	rok	rok
---	-----	-----

1. Osnovna terminologija (IT, IS, DBMS, BP), Oblasti primene IT,		
2. Kategorije problema podesne za rešavanje uz pomoć IT, Oblici primene IT		
3. Evolucija odnosa aplikacije, OS i H/W		
4. Generacije programskih jezika		
5. Generacije H/W		
6. Etape u razvoju sistema za upravljanje podacima		
7. Obrada zasnovana na fajlovima		
8. Obrada zasnovana na korišćenju BP		
9. ANSI-SPARC arhitektura		
10. Kategorije savremenih DBMS		
11. Relacije BP		
12. Multimedijalne BP		
13. Sistemi za upravljanje bazama podataka (DBMS)		
14. DBMS-osnovne funkcije		
15. DBMS-osnovne komponente	Jun-07	
16. DML funkcije		
17. DDL kompajler		
18. Funkcije SQL DB servera		
19. Arhitektura SQL DB servera	Jan-07	Sep-07
20. Izbor arhitekture SQL DB servera		
21. Arhitektura RDBMS Oracle10g	Sep-06	
22. Detaljna arhitektura Oracle10g servera		
23. Oracle10g procesi		
24. Oracle instance		
25. Logička struktura Oracle BP		
26. Fizička struktura Oracle BP		
27. Objektno-relacioni DBMS	Mar-07	
28. Objektno-orientisani DBMS		
29. Modeli podataka		
30. SQL i programski interfejsi	Apr-07	
31. Programski interfejsi BP		
32. Ugnježdjeni API (ESQL API), funkcije, obrada		
33. Obrada ugnježđenog SQL API-ja		
34. DB interfejsi na nivou poziva (CLI API)	Nov-06	
35. Komparacija ESQL API i CLI API		
36. Tipovi (kategorije) IS, aplikacija i korisnika		
37. Front-end alati baze podataka		
38. Osnovi upravljanja procesima		
39. Upravljanje transakcijama		
40. Svojstvo ACID		
41. Kratke i duge transakcije, definicija i komparacija		
42. Upravljanje konkurentnim pristupom	Mar-07	
43. Problemi konkurentnosti (transakcija)		

44. Pesimisti č ki pristup upravljanju konkurentnim radom	
45. Nivoi zaključavanja objekata BP	
46. Upravljanje transakcijama primenom TPM	Sep-07
47. Pojava i razrešavanje dead-lock-ova	
48. Upravljanje sigurnošću u baze podataka	
49. Problemi sa transparentnošću u procesa	
50. Distribuirani sistemi (definicije, prednosti, nedostaci)	
51. Distribuirana obrada, distribuirana baza i distribuirani DBMS	
52. Tehnike distribucije podataka	
53. Horizontalno i vertikalno particioniranje podataka iz BP	
54. Keširanje podataka	
55. Ekstrakt i replika	
56. Replika kod MySQL-a	
57. Korišćenje replikacionih servera	
58. Kanalisanje i TPM	
59. Standardi za TP monitore	Sep-06
60. Komparacija TPM i memorijskih procedura	Apr-07
61. Način distribucije procesa	
62. Udaljeni zahtev	
63. Udaljena transakcija	
64. Distribuirani zahtev	
65. Distribuirana jedinica posla	Jan-07
66. Protokol dvofaznog izvršavanja	Nov-06
67. Osnovna svojstva distribuiranog DBMS	
68. Multidatabase sistemi	
69. Federalizovani MDBMS sistemi	Mar-07
70. Arhitektura federalizovanih MDBMS sistema	
71. Vlasništvo nad podacima	
72. Modeli obrade podataka	
73. Arhitekture informacionih sistema	
74. Centralizovana višekorisnička arhitektura	
75. Distribuirana jednokorisnička arhitektura	
76. Distribuirana višekorisnička arhitektura	
77. Prednosti i nedostaci distribuiranih IS	
78. Arhitekture distribuiranih IS (DIS)	
79. Master-slave model	
80. Klijent-server model	
81. Model od-sloja-do-sloja (P-to-P)	
82. Grupni model	
83. Model distribuiranih objekata	
84. Model multimedijalnog toka	
85. Klasifikacija klijent-server sistema prema hijerarhiji, lokaciji	
86. Klasifikacija klijent-server sistema prema nameni servera	
87. Fajl server	
88. DB server	
89. Transakcioni server	

90. Objektni aplikacioni server	
91. Grupni server	
92. Web aplikacioni server	
93. Komponente distribuiranih IS	
94. Dvoslojna C/S arhitektura	
95. Troslojna C/S arhitektura	
96. Komparacija 2-slojne i 3-slojne arhitekture	
97. Osnovna struktura monolitne aplikacije i alokacija funkcija na elemente C/S sistema	Apr-07
98. Mogu ć a distribuirana rešenja particioniranja aplikacije	
99. Osnovne C/S tehnologije	
100. Tehnološki elementi C/S sistema	
101. Generalizacija s/w arhitekture klijent-server sistema	Sep-06
102. Tehnologije servera (arhitektura s/w servera, h/w platforma)	
103. OS servera	
104. Bazni servisi OS	
105. Prošireni servisi OS	Jan-07
106. Tehnologije klijenta (arhitektura sw klijenta, h/w platforma)	
107. OS klijenta	
108. Korisni ć ki interfejs klijenta	Nov-06
109. Alati za razvoj C/S aplikacija	
110. Struktura i elementi savremenog razvojnog alata	
111. Izbor alata za razvoj C/S aplikacija	
112. Integrisana razvojna okruženja (IDE)	
113. Primeri proizvo đ a ć kih IDE-ova	
114. Primeri otvorenih IDE-ova	
115. MS .NET okvir (Framework)	Sep-06
116. Srednji sloj (middleware) C/S sistema, namena, položaj, servisi	Sep-07
117. Celovite arhitekture srednjeg sloja, sli ć nosti i razlike	
118. OSF DCE	
119. OSF DCE arhitektura +	
120. Distribuirani Naming servisi	
121. Tipovi middleware-a	
122. Srednji sloj RPC tipa	Apr-07
123. Srednji sloj MOM tipa	
124. Transakciono orijentisan MW (transakcioni monitori)	
125. Transakciono orijentisan MW (aplikacioni serveri)	Nov-06
126. MW baziran na distribuiranim objektima	
127. Middleware tipa OMG CORBA	Mar-07
128. Centralni deo CORBA arhitekture	
129. Povezivanje/integracija aplikacija u preduze ć u (EAI)	
130. EAI na nivou procesa	
131. Brokeri poruka	Nov-06
132. Srednji sloj za povezivanje sa BP (namena, položaj i funkcije)	
133. Na ć ini povezivanja klijenta i servera	Mar-07
134. Native/proizvodjać ki DB MW	
135. Arhitektura c/s sistema sa DB MW u homogenom okruženju	
136. Arhitektura c/s sistema sa DB MW u heterogenom okruženju	Sep-06 Jan-07

137. Implementacije tipa zajedničkog interfejsa	
138. Povezivanje sa ODBC, JDBC	
139. Povezivanje sa OLE-DB i ADO	Jan-07
140. MS ADO.NET	
141. Arhitektura ADO.NET	
142. Povezivanje putem zajedničkog gateway-a	Apr-07
143. Povezivanje putem zajedničkog protokola	Jun-07
144. Otvoreni sistemi i IT standardi, standardizacija organizacije	