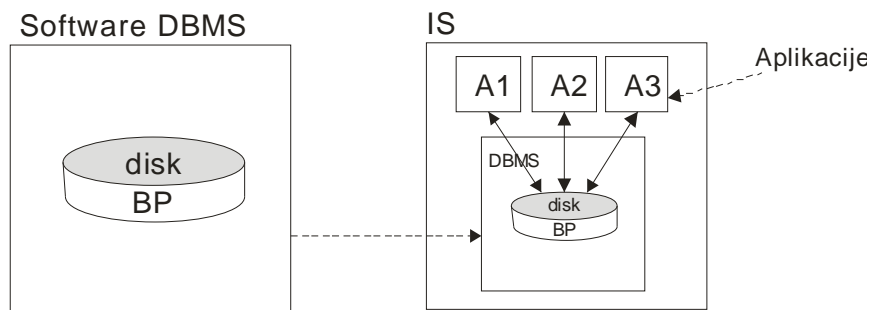


KLIJENT – SERVER SISTEMI

IT –terminologija

- IT- skup metoda i sredstava iz oblasti računarstva i TK(ICT) koji pomažu u prikupljanju, čuvanju, obradi i prenosu informacija.
 - ▶ ICT - Information Communication Technologies
 - ▶ TK - Telecommunications
- IS- Informacioni sistem
Sistem mu kome se veze između objekata i veze sistema sa okolinom ostvaruje razmenom informacija.
- DBMS – je s/w sistema za čuvanje i korišćenje podataka, uz logičku i fizičku 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 zajednički koriste svi procesi obrade (Aplikacije) u sistemu.
 - ▶ DBMS != BP



Motivacija

Značaj IS

- Velike investicije u IT, pre svega u IS.
- Značaj IS za svakodnevno poslovanje
- Današnji IS su uglavnom DISTRIBUIRANI
- c/s arhitektura je dominantna danas u distribuiranim sistemima
 - ▶ c/s - Klijent-Server

IT Domen

- Značajne investicije u SAD u nivo računarske sisteme, preko 200 GUSD godišnje, poslednjih 5 godina, slično u Zapadnoj Evropi.
GUSD= Giga USD tj.10⁹
- Procena IT u Jugoslaviji ulagano oko 200 MDEM/god.
- Prodaja RDBMS je 15 – 20 GUSD/god., a 50 GUSD/god. računajući alate

Lična motivacija

plate u 2002.

- SAD - 48.000 USD-god. BRUTO (ing. početnik)
- 96.000 USD - srednja vrednost
- Japan - 62.900 USD
- UK - 57.800 GBP (samo 25% za Induse) BT British Telecom
- Istočna i Južna AZIJA - 13.300 USD/god.
- SCG - Širok raspon 300-1000 E

Oblasti i oblici primene IT

Kategorije problema podesne za rešavanje na računaru:

- Finalni rezultat mora biti ekonomski opravdan
- Jasno definisan i sa dostižnim ciljevima
- Repetitivan
- Količina podataka značajna (posl. apl.)
- Brojni / Složeni proračuni (Naučno-Teh. apl.)

Različite oblasti primene (Finansijske institucije, ...)

Oblici primene –Brojne moguće klasifikacije

- Ugrađeni s/w (embedeed s/w) u uređaje
- Merni sistemi
- CAD / CAN / CIM (Compiter Integrated manufacturing)
- Informaciono-Upravljački Sistemi
- Simulacioni Sistemi
- Simulaciono-Trenažni Sistemi
- Informacioni Sistemi, razni:
 - poslovni, tehnički, MIS, DMS, GIS, EIS, DSS, IIS
 - MIS - Management IS
 - GIS - Geographic IS
 - EIS - Enterprise (za kompaniju) IS, Executive (za direktore) IS
 - DSS - Decision Support System
 - IIS - Intelligent IS

Istorijski pregled razvoja IS

Razvoj HARDWARE-a

- Savremeno Računarstvo postoji oko 60 god.
- Organizacija, Funkcije i Performanse su se menjale u skladu sa promenom tehnološke osnove
- Razvoj računara se obično prati kroz generacije
 - I. 1942.-1950.
 - II. 1959.-1965.
 - III. 1965.-1970.
 - IV. 1970.-1980.
 - V. 1982.-1990.

I.generacija: 1942.-1950.

- ▶ Elektronske (vakuumske) cevi
- ▶ Feritne memorije
- ▶ Magnetni doboš
- ▶ Mašinski jezik

PRIMERI: ENIAC 1945., UNIVAC I 1951. (30 tona), IBM 650 1953. (masovna proizvodnja).

II.generacija: 1959.-1965.

- ▶ Poluprovodnički elementi
- ▶ Magnetne trake
- ▶ Mašinski + Simbolički Jezik (Assembler)

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 (veći stepen integracije)
- ▶ Virtualna memorija
- ▶ Rast Brzine

PRIMERI: IBM System/360

V.generacija: 1982.-1990.

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

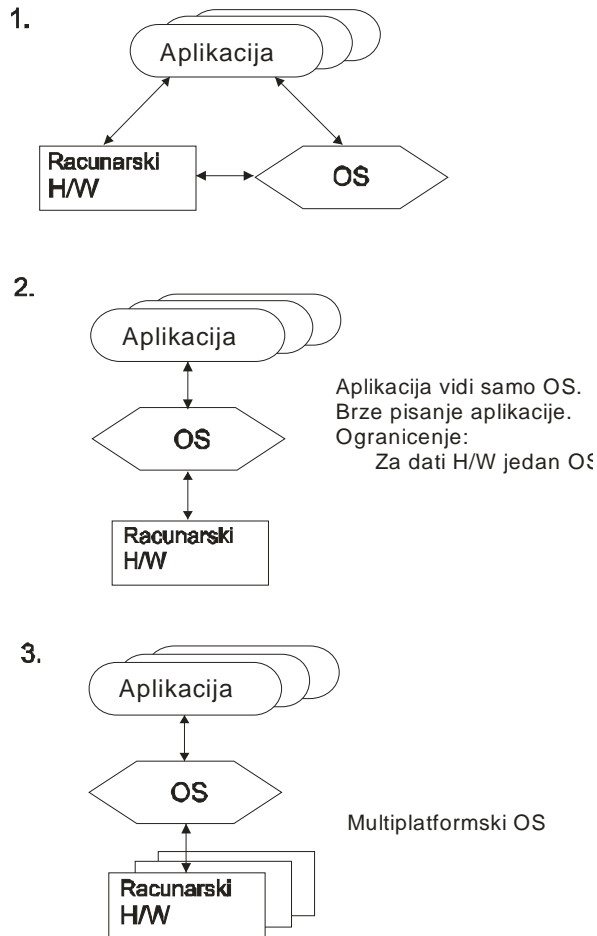
U paraleli 70-tih.

- ▶ Pojava MIKROPROCESORA (8-bitni)
- ▶ Pojava MIKRORAČUNARA
 - Apple II 1977. Mikroračunar (CPU μ P, Monitor, Keyboard, Floppy)
 - IBM PC 1981. (Intel 8088, 16-bit, 2MHz)

- ▶ Od tada nezaustavljiv razvoj PC-a

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

Evolucija odnosa Aplikacije, OS-a i h/w PC-a



Korišćenje μ P početkom 80-tih

- ▶ Radne Stanice, Mini Računari HP, DEC,... za Gornji nivo tržišta (UNIX, grafika)
- ▶ Personalni Računari, donji nivo tržišta, (povećanje personalne produktivnosti)
- ▶ Kontinualni rast (32, 64-bit) pojava portabilnih OS, PCI magistrale
- ▶ Konvergencija WS i PC-a po performansama i ceni

Ilustracija:

1981. prodato 13.000 IBM PC-a
 1995. prodato 60.000.000 PC-a
 2001. prodato 300 Gkom IC(čipova)
 2003. prodato 360 Gkom IC(čipova) po pros. ceni 0.5 USD/kom. = 160 G USD

- Magnetni doboš 1947/48., Mag. diskovi 1956.
- Feritna memorija 1951., Magnetna traka 1975.
- Linijski printer 1954., Laserski 1964.
- Monitor 1960., miš 1964.
- Floppy 1970., CD ROM 1984.

Poenta

- Mnogi elementi IT i koncepti postoje odavno.
- Dugačak inkubacioni period (10-15 god.) od pojave do široke primene.

Razvoj Softvera- Programski Jezici

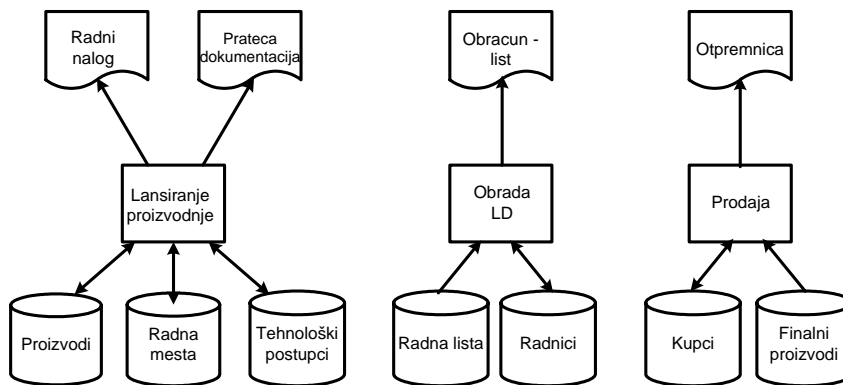
- I. generacija(1GL): 45.-60. Mašinski+Simbolički Jezik (Asembler)
- II. generacija(2GL): 55.-75. Visokog nivoa, Proceduralni, (Fortran 55/77/90, Cobol 59, Algol 60, Basic 64)
- III. generacija(3GL): od 65. na dalje. Jezici opšte namene C 70. Pascal 71., Ada 80./95.,...) specijalni (Lisp, Prolog), OOJ (C++ 83., Java 95.- Platform Independent)...
- IV. generacija(4GL): od 75. Proceduralni i Neproceduralni, Upitni (SQL), generatori koda (CASE), DS jezici
Proceduralni: Specificira se način na koji dolazimo do rešenja.
Neproceduralni: Zadaje se samo željeni rezultat.

OS:

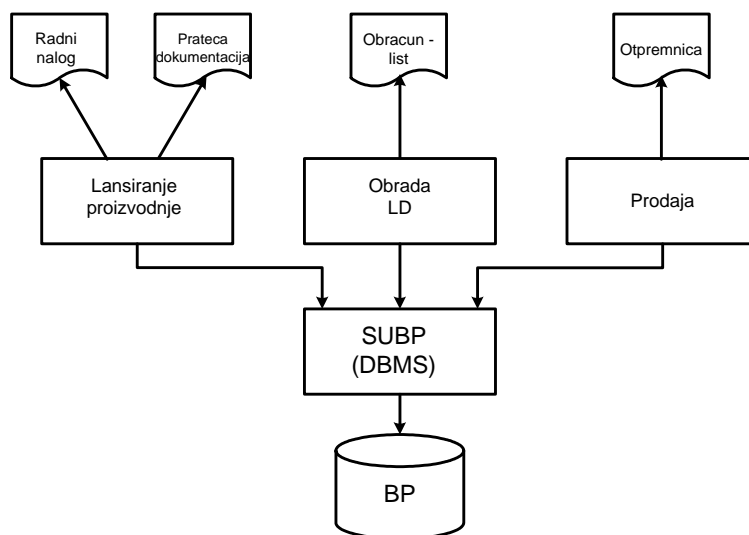
- IBM OS 360 1964.
- DEC VMS 1978.
- UNIX(Bell labs) 1970.
- OS IBM PC, PC DOS 1980.
- Ms Windows 1.0 1985., Win 3.0 1990., Win 95., Win 98., Win NT, Win 2000, Win XP,...
- Razni UNIX-i (HP-UX, AIX, Solaris),Linux,...

Razvoj s/w komunikacija

- ➔ e-mail 1982. (komercijalno), prva poruka 1971.
- ➔ www (URL, HTTP, HTML) 1990.
- ➔ Browser (Netscape) 1994.
- ➔ XML Spec. 1995., finalizovano 1998.
- ➔ Brzi rast GLOBALNE MREŽE – Internet-a.

Klasična obrada podataka – Primer:**Računarska obrada podataka – Primer:**

IS oslonjen na BP



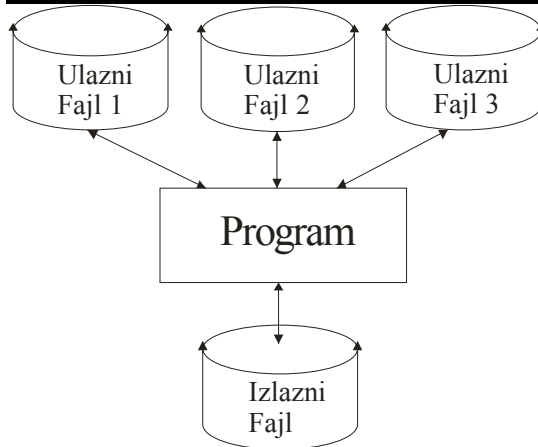
Upravljanje podacima

- ➔ Etape u razvoju sistema za upravljanje podacima
- ➔ Kategorije savremenih DBMS
- ➔ Sistemi za upravljanje BP (SUBP)

Etape u razvoju sistema za upravljanje podacima

- Programsko upravljanje zapisima
- On-Line (HIJERARHIJSKE / MREŽNE) BP, Hijerarhijski/ Mrežni model
- Relacione BP
- Multimedijalne BP

Programsko / sekvencijalno upravljanje zapisima (1955-1970)



U osnovi je problem koji se rešava. Najčešće Finansijski problemi.

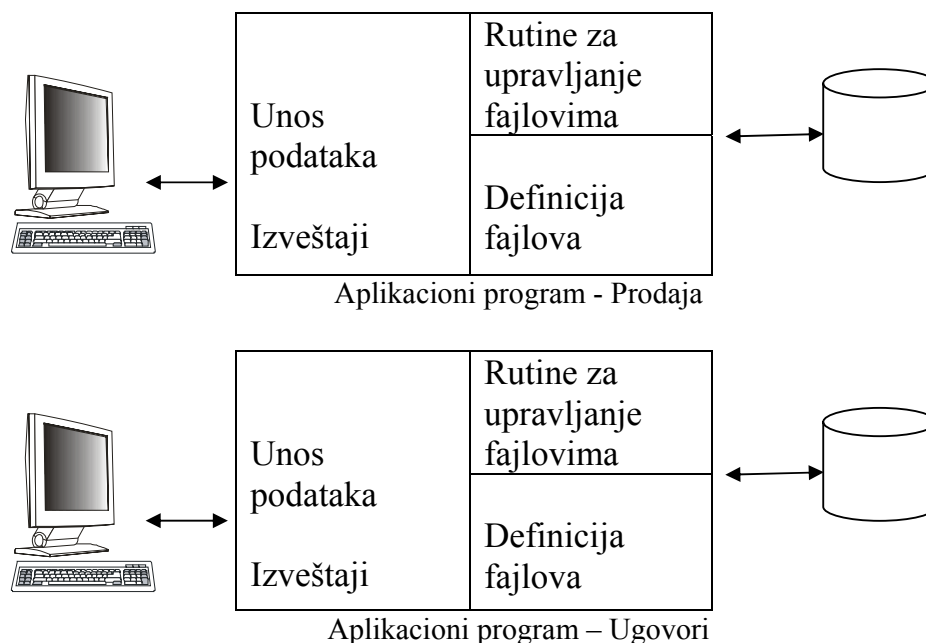
Ulazni podaci su organizovani u fajlove, program ih obrađuje i daje rezultat u izlaznom fajlu. Zapisima se upravljalo sekvencijalno zbog tehnologije Magnetnih Traka. Ovakav koncept se i danas primenjuje za probleme malim brojem ulaznih podataka.

Loader- u BP upisuje podatke iz fajlova.

Izlazni Fajl – **Master fajl**.

Ovaj koncept je i danas primenjiv u Bankarskim sistemima.

Tradicionalna obrada zasnovana na fajlovima



Ograničenja tradicionalnih – Fajlovski organizovanih sistema

- ➔ Razdvojenost i izolovanost podataka
Svaka aplikacija ima svoj ulaz i izlaz
- ➔ Dupliciranje podataka
Problem održavanja višestrukih podataka.
- ➔ Zavisnost podataka od fizičke strukture
- ➔ Nekompatibilnost fajlova
- ➔ Fiksni upiti
- ➔ Proliferacija apl. programa
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.

Novi pristup:

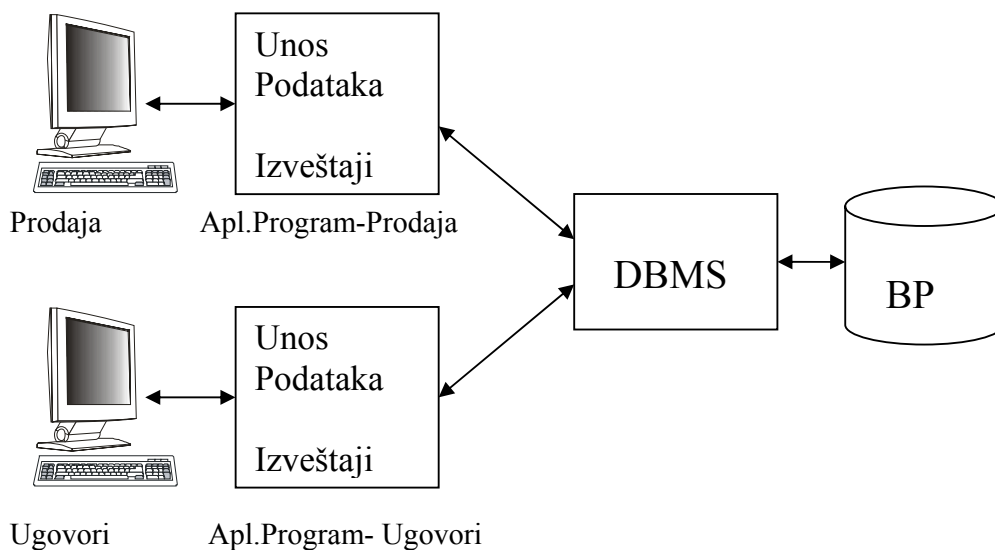
- BP
- DBMS

Poništava individualne fajlove.

Prevazilaze se ograničenja I etape.

Koncept BP traje i danas.

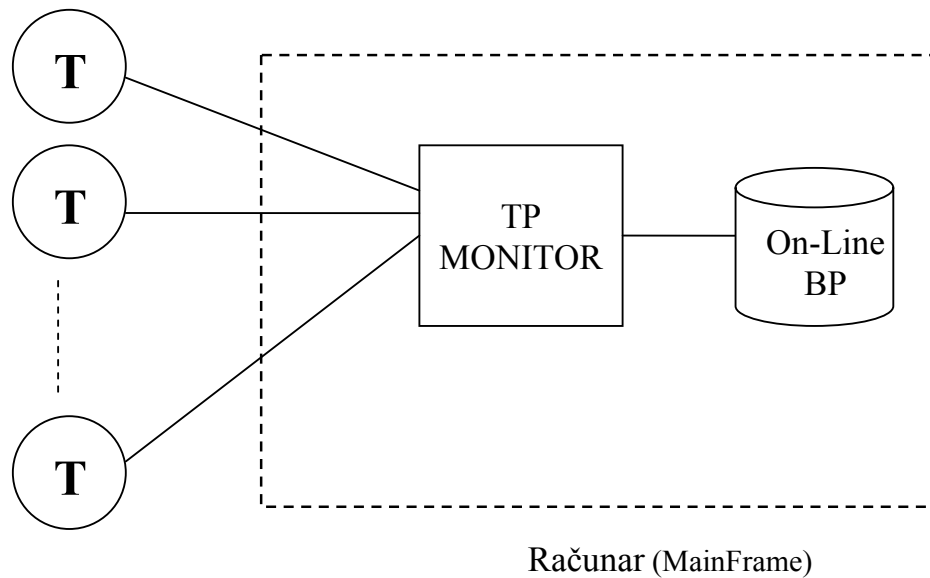
Obrada zasnovana na korišćenju BP



Nema pristupa BP bez prolaska kroz DBMS.

II. etapa: On-Line IS (1965.-1980. i kasnije)

Oslonjen na Hijerarhijske / Mrežne BP

**TP** – Tele Processing (Daljinska Obrada)**MF** – Main Frame**T** – Terminal

Obrada se vrši u Main Frame računar.

TP Monitor ima ulogu multipleksera za razmenu podataka između periferije i Glavnog dela obrade.

Sistem je ON-LINE da bi se ukazala razlika i pšomak u odnosu na I. etapu. U prvoj etapi nismo imali užurno stanje sem na nivou fajla.

Značaj u sistemima za rezervacije (System Amadeus).

Danas, zbog porasta broja računara uvezi se prave replikacije BP (SnapShot), pa sa tom kopijom rade Terminali a u toku noći se vrši ažuriranje. (BATCH obrada – Paketna Obrada)

Hijerarhijski i Mrežni DBMS

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).

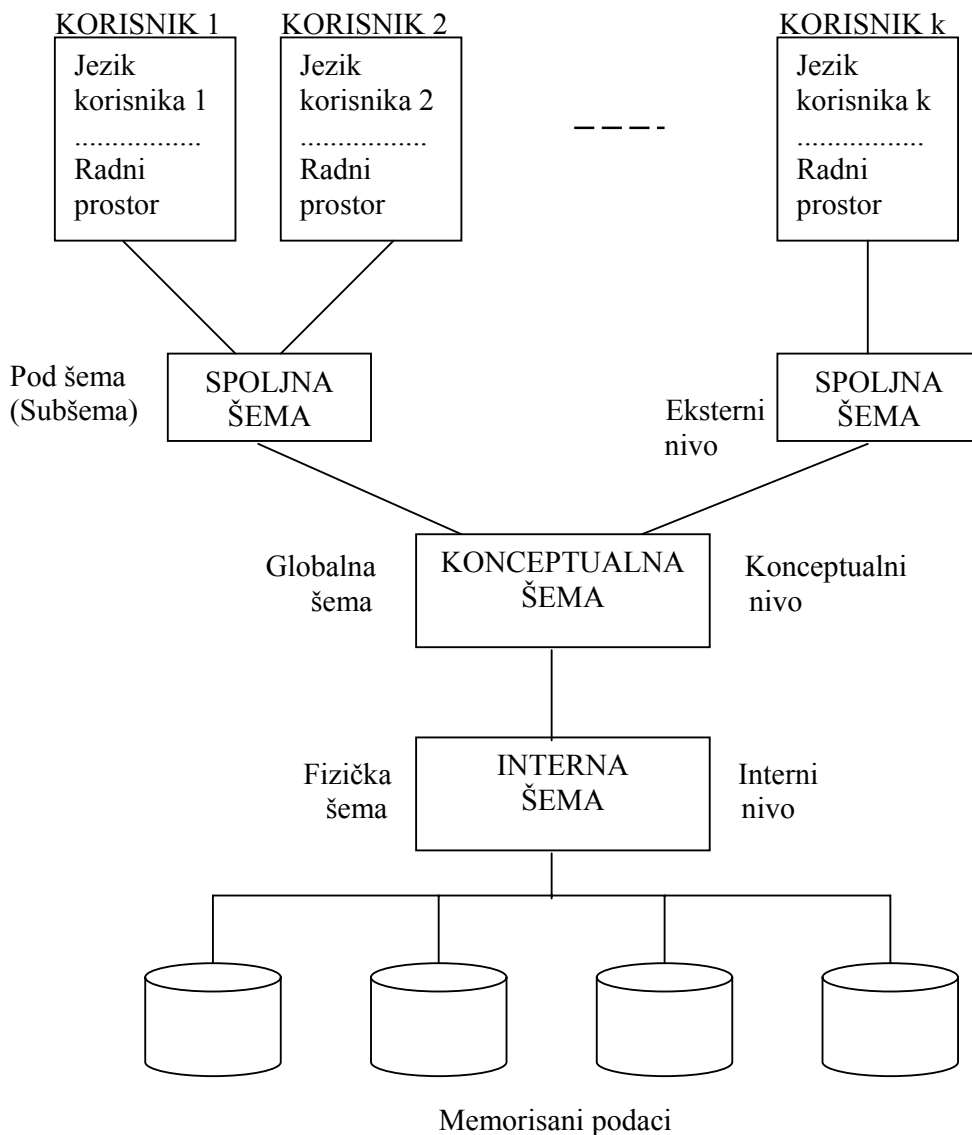
• Mrežni model

- ▶ 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.

Značajni Koncepti

- Prvi predlog standardne terminologije i arhitekture DB sistema dati 1971. od strane **CODASYL / DBTG**, **dvo-nivovski** pristup:
 - Sistemski (Šema)
 - Korisnički (Subšema)
 - Šema predstavlja sve podatke u BP – Sistemski Pogled.
 - Subšema je pogled na BP iz aplikacije – Aplikativni pogled.
 - ▶ **CODASYL** – Conf. on Data System and Languages /
DBTG – DataBase Task Group
definisala predlog standarda za tri jezika:
 - **DDL** šeme – Data Definition Language
Spec. programski jezik kojim definišemo podatke, pomoću njega projektujemo BP.
 - **DDL** Subšeme – Jezik koji koristi aplikacioni programer.
 - **DML** – Data Manipulation Language
Jezik krajnjih korisnika.
 - ➔ Brojne implementacije, tzv. **CODASYL / DBTG** sistemi.
 - ▶ **ANSI** nije usvojio ovaj standard.
 - **ANSI / x3 / SPARC** (Standards Planning and Requirements Committee)
predložio 1975.
 - ▶ **Tro – nivovska** arhitektura sa sistemskim katalogom, koji omogućava Logičku i Fizičku nezavisnost podataka, što je ključni koncept BP.
 - Subšema, Logička Šema, Fizička Šema.
 - Koncept Transakcija i Zaključavanja.
- Primeri: IBM IMS, UNIVAC DMS 1100, TOTAL, ADAPASE, IBM DL/I.
- IMS – Information Management System.

ANSI / SPARC arhitektura

- Konceptualna šema – Šta sve treba da sadrži BP. Pravi je DBA (Data Base Administrator).
- Podšema – korisnikov pogled na BP, reprezentacija podataka.
- Interna šema – Organizuje podatke na fizičkom medijumu.
- Eksterni nivo – korisnik vidi BP kroz svoju Subšemu.
 - ▶ Između Konceptualne šeme i Interne Šeme obezbeđuje se nezavisnost Aplikacije od Fizičke šeme tj. Fizičke organizacije podataka.
 - ▶ Promene u Konceptualnij šemi ne treba da utiču na aplikaciju.
 - ▶ Postoji Logička nezavisnost aplikacije.
 - ▶ Ako se doda podatak u BP, aplikacija se ne menja.

Kategorije savremenih DBMS

- Relacioni sistemi za upravljanje BP – RDBMS
- Objektno – Relacioni sistemi za upravljanje BP – OR DBMS
- Objektno – orjentisani sistemi za upravljanje BP – OO DBMS

Savremeni DBMS-ovi

Kategorizacija nije jednostavna, ovde je diferencijacija izvršena na osnovu:

- Modela podataka
- Upitnog jezika
- Računskog modela / navigacija pristupom.

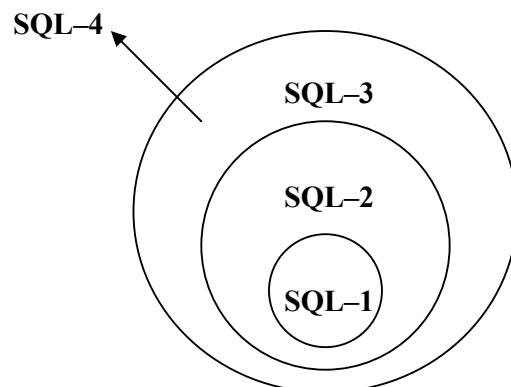
Relacione BP

- Entiteti i relacije prikazani na uniforman način, putem TABELA.
- BP – TABELA – REKORDI (redovi) – ATRIBUTI (kolone)
- Svaka kolona ima tip podatka, ograničen broj (tipova podataka)
- Relacije implicitne – Spoljni KLJUČ
- Jedinstven jezik za Definisanje podataka, navigaciju i manipulaciju (SQL)
- Standardizovan Strukturirani Upitni Jezik (SQL) – Neproceduralan.
- Odgovarajući Standardi: ANSI, ISO
 - ▶ SQL-1: 1986./1989., ANSI-89
 - ▶ SQL-2: 1992.
 - ▶ SQL-3: 1999. Nov, specifikacija 2000 strana, malo ko još usklađen.

Proizvođači RDBMS-ova treba da usklade svoje proizvode sa standardima

- **Stored-Procedure** (stoje u BP i pokreću se kroz katalog)
Smanjuje se vreme pristupa, duže je kad su procedure u aplikaciji.

- ➔ Navedeni Standardi imaju međusobni odnos:
Svaki sledeći je NADSKUP prethodnog.



Standard precizira šta SQL treba da ima, a ne i kako to radi.
Razvoj ide ka SQL-4 standardu.

Relacione BP

- Originalni razvijena od strane **E.F.Codd**–a 1970. (matematički fundirane)
- Programi 5–10x jednostavniji od navigacionih.
- Podaci se pretražuju na bazi vrednosti u Polju.
- Pregled rezultata Upita se vrši pod kontrolom kursora.
- Ograničen broj podržanih tipova podataka:
 - ▶ **char,string,number,time,date,currency**
- ➔ **Primeri:** Oracle 7, MS SQL Server, DB2, Sybase Sys. 10/11, Informix, Ingres..., preko 100 realizacija

Multimedijalne BP (1995. na dalje)

- Multimedijalni podaci:
 - ▶ Tekst, Grafika, Slika
 - ▶ Audio, Video, njihova kombinacija
 - ▶ Bogati informacijama, Dimenzije! (zauzeće memorije)
- Ekspanzija Multimedijalnih podataka (Internet doprineo)
- 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).
 ER DBMS–Extended Relational DBMS
- **Objektno orjentisani** sistemi za upr. BP (OO DBMS).

Objektno–relacioni sistemi za upravljanje BP

- Pokušaj ujedinjenja Relacionih i Objektnih BP:
 - ▶ Proširene–Relacione (ER) ili Objektno relacione (OR)
- Koristi model podataka koji BP dodaje objektnu orjentisanost.
- Sva perzistentna informacija je u tabelama, ali neki elementi tabela imaju bogatiju strukturu, ADT (Abstract Data Types):
 - ▶ Tekst, Slika (Image), Grafika, Animacija
 - ▶ Audio, Video, georeferencirani
 - ▶ Vremenski markirani
- Važno: **Objekti != Tipovi podataka**
OR DBMS != OO DBMS
- Ne postoji Enkapsulacija procedura (metoda) sa podacima, ograničena podrška drugim OO svojstvima
- Podržavaju prošireni oblik SQL–a (Object SQL), uključuje:
 - upite sa ugnjeđenim Objektima, atribut sa skupom vrednosti,
 - uključivanje metoda i iskaza pretraživanja koji uključuju i elemente ADT.
- Osnovni interfejs prema BP je i dalje SQL sa ekstenzijama (SQL–3, SQL–4 u razvoju)
- 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)

Obj. Orjentisani sistem za upravljanje BP (OO DBMS)

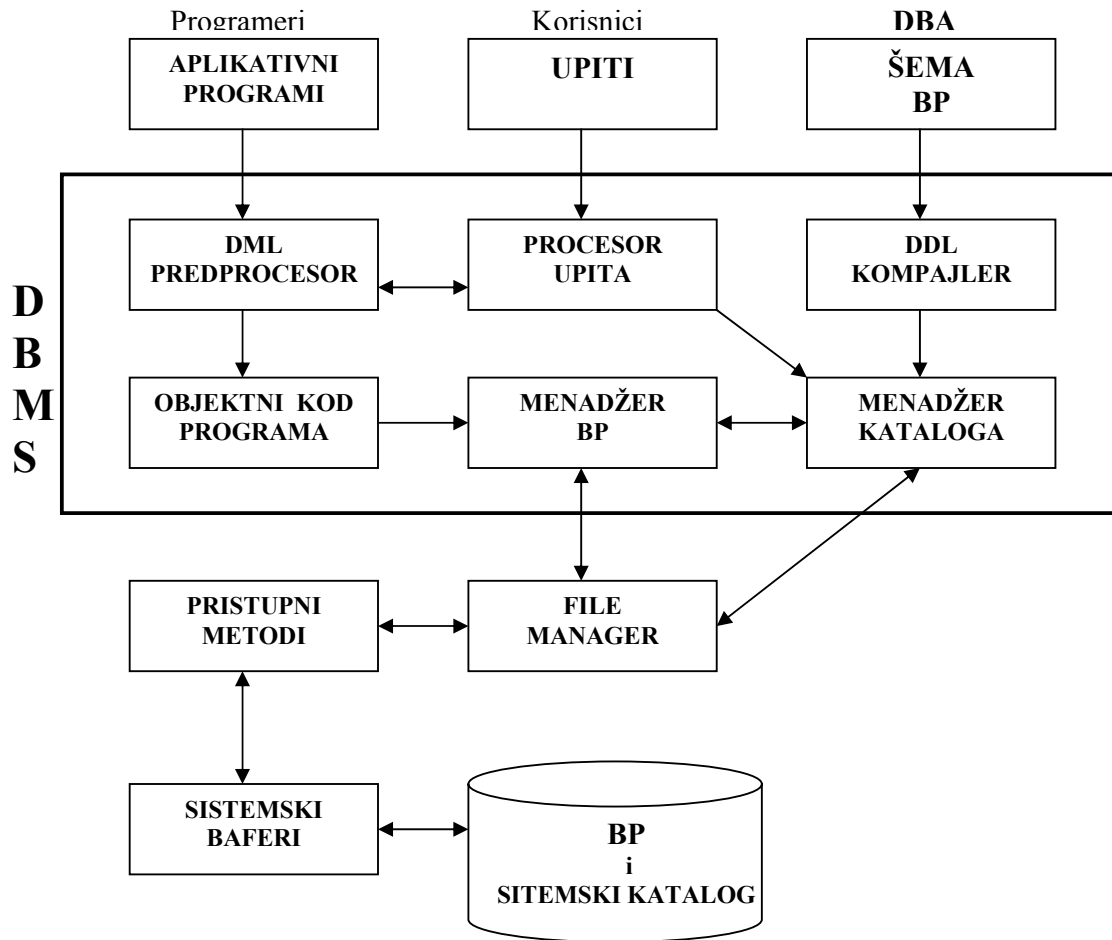
- Ne postoji oficijelni standard za Obj. BP, **de-facto** standard je **ODMG v.2.0** definisan od strane Object Database Management Group
- Objektne BP koriste model podataka koji ima OO koncepte– klasa koja obezbeđuje identifikatore objekata (OID) za svaku perzistentnu instancu klase, enkapsulaciju, višestruko nasleđivanje i podržava ADT.
- Objektna BP proširuje funkcionalnost Obj.Orj. Programskih Jezika (C++, Java, SmallTalk...)
- ➔ Mali broj proizvođača i korisnika Obj. BP.
- Objektno orjentisani jezik predstavlja Jezik kako za aplikaciju tako i za BP.
- ODBMS je do sada integrisan sa: C++, C, Smalltalk, Java, Lisp–om
- ODMG– 1993. definiše i deklarativni jezik OQL za upit DB objekata, OQL nije 100% kompatibilan sa SQL–om.
- Većina objekata BP podržava i SQL putem ODBC–a.
- U okviru ODBMS–a primarni interfejs za kreiranje i modifikaciju objekata je direktno putem Objektnog Jezika, korišćenjem Nativne sintakse jezika.
- Svakom objektu se automatski dodeljuje jedinstven, nepromenljiv identifikator–OID
- Podesno za CAD/CAM sisteme.
- ➔ Primeri: Gemstone, jasmine(CAI),Itasca (Ibex obj. sys), Poet v.5.0, O2 (Ardent s/w), Objectivity/DB, Ontos DB,...

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,...

Funkcije DBMS–a

- Memorisanje, pretraživanje i ažuriranje podataka
- Katalog dostupan korisniku
- Podrška Transakcijama
- Upravljanje konkurentnim pristupom
- Servisi Oporavka (Recovery)
- Servisi autorizacije
- Servisi integriteta
- Podrška komunikaciji
- Uslužni servisi

Struktura DBMS-aKomponente DBMS-a:

- DDL kompajler
- Menadžer kataloga
- Procesor upita
- DML Predprocesor
- Menadžer Baze Podataka
- Fajl Menadžer

DDL kompajler vrši konverziju DDL instrukcija u skup tabela koje sadrže meta-podatke. Te tabele se memorišu u Sistemsom 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 fizičkim U/I podataka nego prosleđuje zahteve za odgovarajućim Pristupnim metodama koji ih U/I u Sistemski Bafer ili Keš (Cache) bafer.

Upravljanje podacima

- Funkcija DB-servera
- Arhitektura SQL DB-servera
- Arhitektura RDBMS Oracle 8

Funkcije SQL-servera

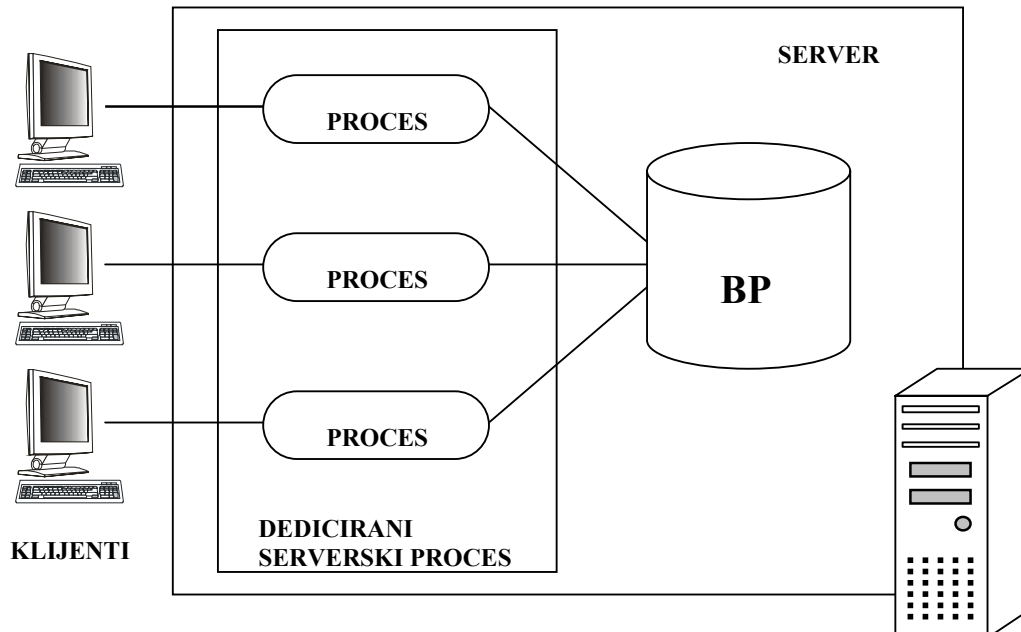
- Od DB-servera C/S aplikacija zahteva podatke i sa njima povezane servise (sort,...).
- DB-server se ponekad naziva “SQL-engine” obezbeđuje siguran pristup deljenim podacima i odgovara na zahteve klijenata.
- On upravlja kontrolom i izvršavanjem SQL komandi. Obezbeđuje logičke i fizičke pogleda na podatke i generiše optimizovane planove pristupa za izvršavanje SQL komandi.
- 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. Većina SQL servera obezbeđuje funkcionalnost na nivou SQL’89. Većina uključuje i neka svojstva SQL’92. Samo nekolicina obezbeđuje proizvođačke verzije mem. procedura , trigeri i pravila (SQL-3)
- Šta je SQL DB server?
 - ▶ Mešavina standardne 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

Arhitektura SQL DB servera

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

- Arhitektura tipa **proces-po-klijentu**
- Arhitektura sa **više niti**.
- **Hibridna** arhitektura

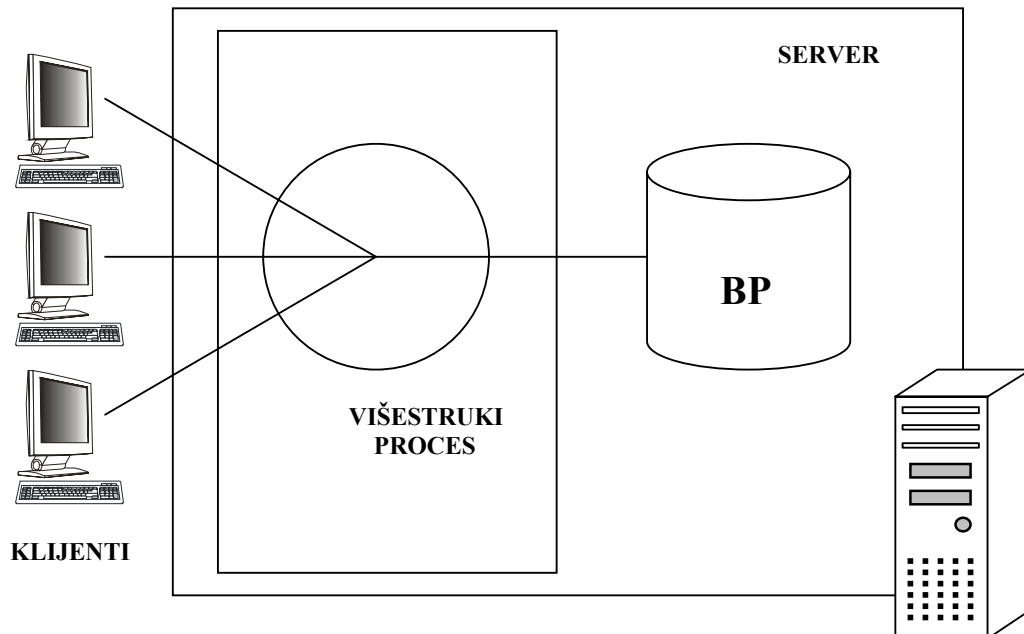
Arhitektura tipa: proces-po-klijentu



/Dedicirani–Dedicated– Namenski/

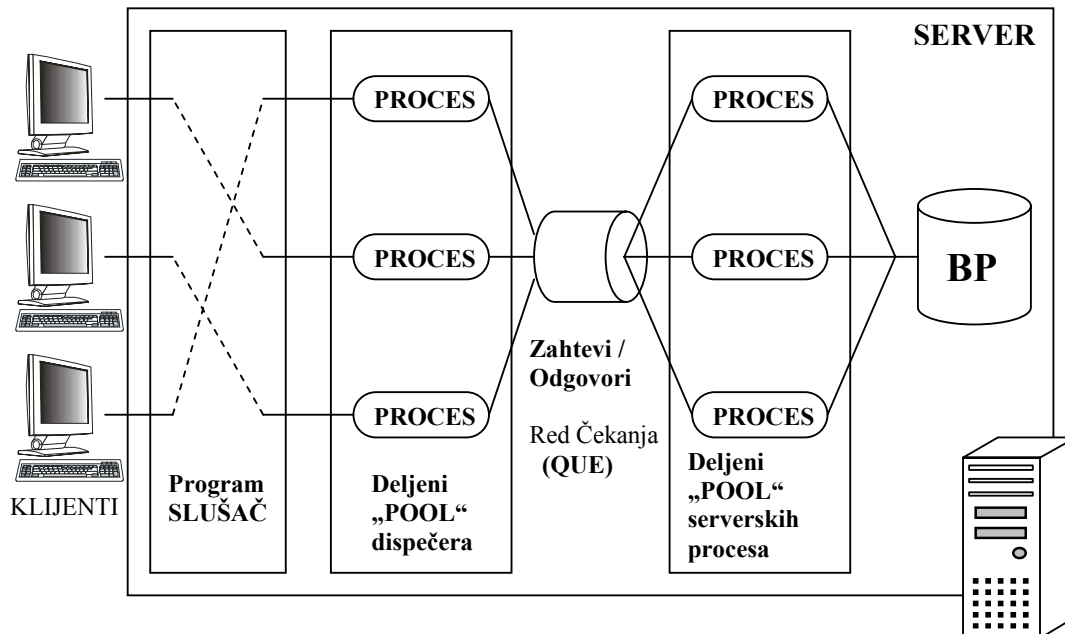
- Obezbeđuje maksimalnu sigurnost dajući svakom klijentu BP sopstveni adresni prostor. BP radi u 1 ili više odvojenih pozadinskih procesa
- ➡ Primeri: Informix, DB2, Oracle 6.
- **Prednosti:** štite korisnika jedan od drugog. Štiti DB menagera od korisnika. Procesi mogu lako biti pridruženi različitim 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 čestih promena konteksta procesa (Prevazilazi se TPM).

Arhitektura sa vise niti



- Obezbeđuje 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.
- ➡ Primeri: Sybase, MS SQL Server
- **Prednosti:**
 - ▶ čuva memoriju i CPU resurse pošto ne zahteva česte promene konteksta.
 - ▶ Serverske implementacije su i nešto portabilnije
- **Nedostaci:**
 - ▶ Korisnička apl. Koja se „**loše**“ ponaša može „**oboriti**“ ceo DB server i njegove taskove.
 - ▶ Korisničke apl. Koje se sastoje od taskova dugog trajanja (dugački upiti) mogu zauzeti resurse servera.
 - ▶ „**Preemptive**“ raspoređivač koji obezbeđuje DB server je po pravilu slabiji od **schedulera** nativnog OS.

Hibridna arhitektura



Sastoje se od 3 ključna elementa:

- Više nitne mreže “**slušača**” koji ostvaruju početnu konekciju tako što klijenta pridružuju dispečeru
- Dispečerskih taskova koji poruke smeštaju u interni red poruka (**message que**) i iz njega šalju odgovore klijentima
- Severskih, deljenih, radnih procesa, koji se mogu ponovo koristiti, koji uzimaju posao iz reda čekanja, izvršavaju ga i smeštaju u izlazni Que.
- ➔ Primeri: Oracle 7, Oracle 8,
- **Prednosti:**
 - ▶ Ove arhitekture obezbeđuju zaštićeno okruženje za izvršavanje korisničkih taskova, ali bez pridruživanja stalnih procesa svakom klijentu.
- **Nedostaci:**
 - ▶ Kašnjenja u redu čekanja.

Izbor najbolje arhitekture – uputstva

- 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 vazna svojstva DB (SQL) servera
 - ▶ Memorisane procedure (store proc.)
 - ▶ Trigeri(okidaci)
 - ▶ pravila

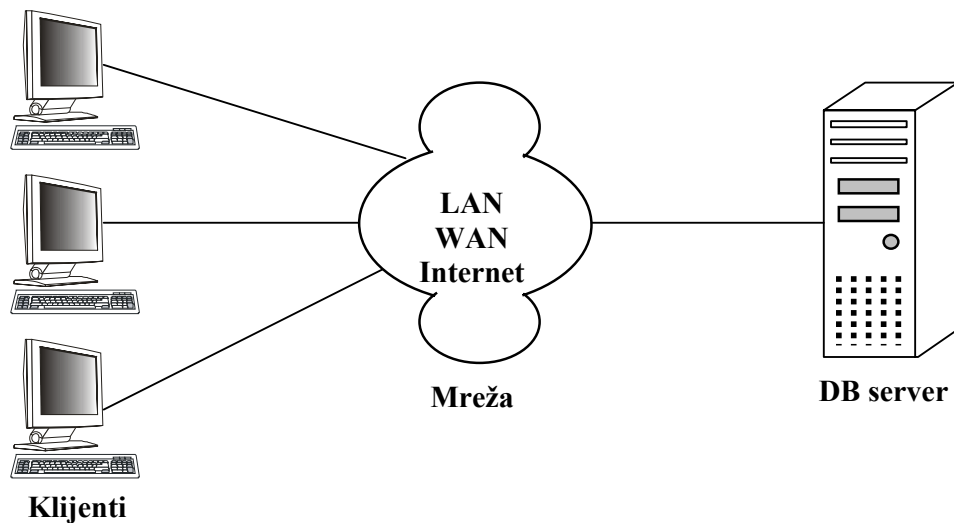
Arhitektura RDBMS Oracle 8

- Hronologija razvoja
- Arhitektura C/S sistema na bazi Oracle
- Arhitektura Oracle 8 servera
- Logicka I fizicka struktura
- Oracle instanca
- Pozadinski procesi

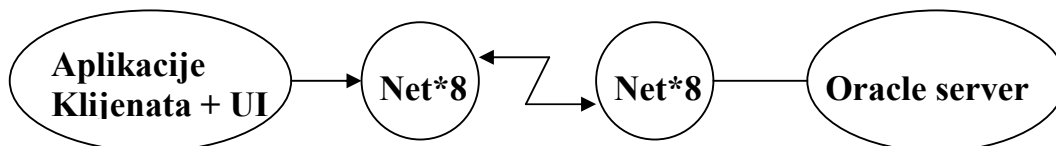
Hronologija razvoja RDBMS Oracle

- Kompanija Oracle: 1997. (C, SQL)
- Oracle 2: 1979. prvi komercijalni RDBMS
- Oracle 3: 1983. na PC-ju
- Oracle 5: 1983. c/s sistem (+SQL*Net –software za povezivanje)
- Oracle 8: 1997. OR RDBMS (universal server)
- Oracle 8i: 1999. Java VM, XML
- Oracle 9i: 2000. Internet, e-business
- Oracle 10g:

Arhitektura hardvera C/S sistema



Arhitektura softvera C/S sistema

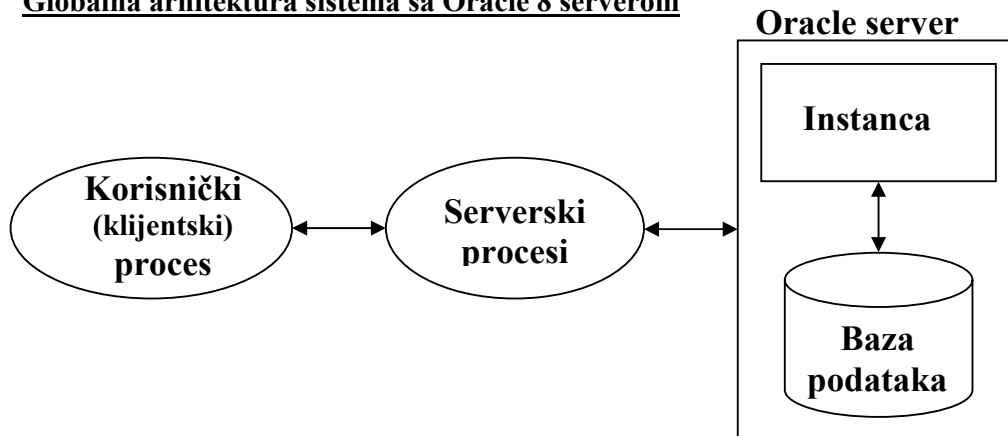


UI – User Interface

Arhitektura sistema sa Oracle 8 serverom

- Oracle server predstavlja OR sistem za upravljanje BP nastao objektnim proširenjem relacionog modela.
- Sastoji se od BP (sa pod. kao i sa kontrolnim i log datotekama) i instance (procesi i sistemska memorija na serveru), koja se može konektovati samo na jednu BP.
- BP poseduje logičku i fizičku strukturu
- Korisnički proces (apl. Program → SQL instr.)
- Serverski proces izvršava SQL instrukcije dobijene od korisničkog procesa.

Globalna arhitektura sistema sa Oracle 8 serverom



Logicka struktura Oracle BP

Pojmovi:

- Prostor tabela (table space)
- Šema
- Blok podataka
- Ekstent
- Segment
- Prostor tabela (table space)
 - ▶ U cilju grupisanja povezanih logičkih struktura Oracle BP je podeljena u logičke memorijske jedinice npr. svi aplikacioni objekti
 - ▶ Svaka Oracle BP sadrži Table space koji se zove SYSTEM, automatski se formira, sadrži tabele Sistemskog kataloga (Rečnik)
Dobro je da postoji još jedan Table Space za korisničke podatke.
- Korisnik (user) ime def. u BP koji se na nju može konektovati i koji može pristupati objektima (podacima)
- Šema predstavlja imenovanu kolekciju objekata šeme (tabele, pogledi, klasteri i procedure)
- Blok podataka predstavlja najmanju jedinicu memorije koju Oracle može koristiti ili alocirati. Definiše se kod kreiranja. Predstavlja multipl veličine bloka u sklopu OS.
- **Ekstent** predstavlja orđedeni blok susednih blokova podataka
- **Segment** (viši nivo organizacije) prestavlja skup ekstenata dodeljeniih određenoj logičkoj strukturi (npr. Podaci svake tabele se pamte u svom posebnom segmentu podataka). Indexi u svojim posebnim index-segmentima.

Fizicka struktura Oracle BP

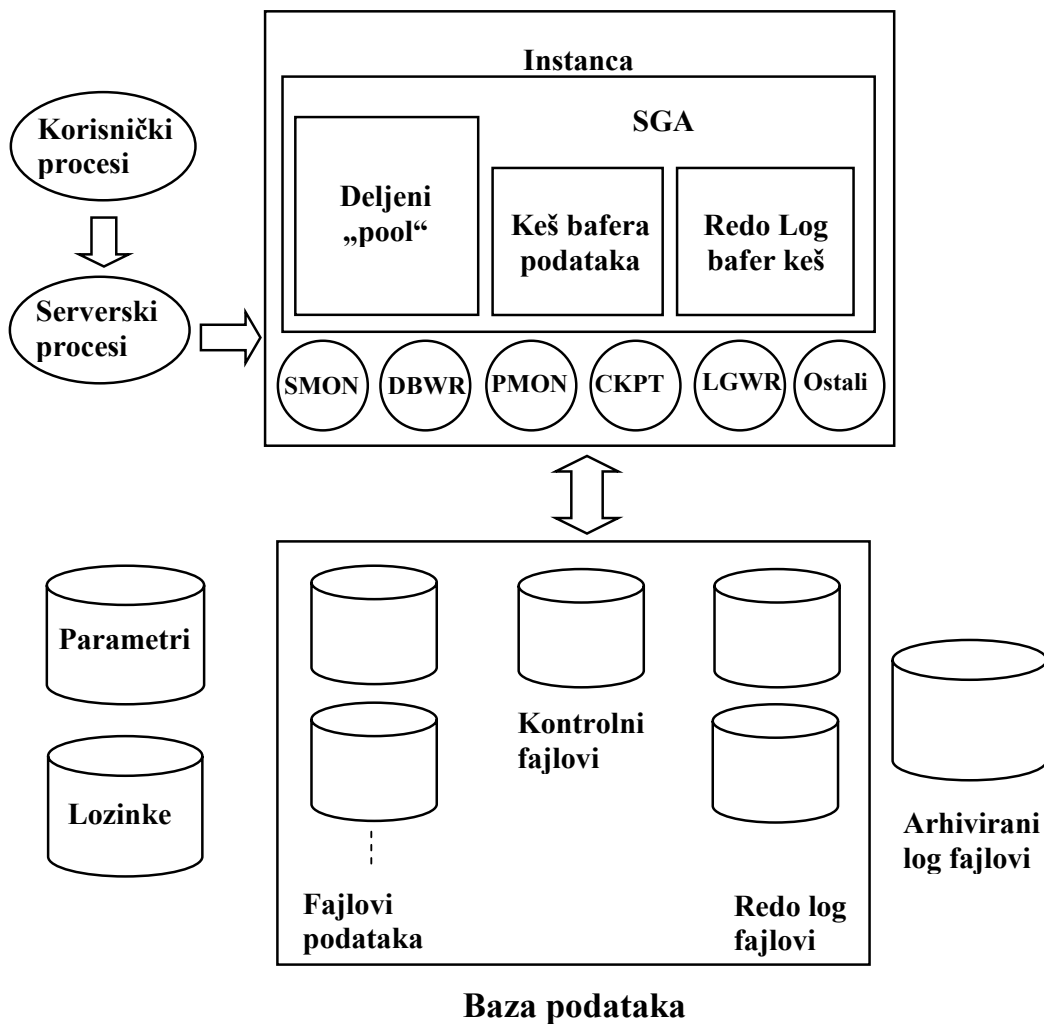
Prestavlja se skupom fajlova (OS), pri cemu se Oracle BP sastoji od 3 tipa fajlova:

- Fajlovi podataka (Data files)
 - Fajlovi sa listom akcija za obnavljanje (Redo Log files)
 - Upravljački fajlovi (control files)
-
- Fajlovi podataka (data files) sadrže stvarne podatke u BP. BP sadrži barem 1 data fajl, pri cemu fajl može biti asociran samo sa jednom BP.
 - Svaka Oracle BP raspolaze sa najmanje dva Fajla sa listom akcija za obnavljanje (redo log) koji registruju sve promene izvedene nad podacima.
 - Upravljacki fajlovi sadrže listu svih fajlova koji cine BP, sa ciljem održavanja i verifikacije integriteta BP. Baza zahteva postojanje barem 1 kontrolnog fajla (Preporuka, više kopija).
 - Koriste se i drugi fajlovi (parametri instance, password, arhivski Redo Log fajlovi, itd...)

Oracle instanca

- Predstavlja kombinaciju memorijskih struktura i pozadinskih procesa.
- Da bise pristupilo podacima u bazi instanca mora biti pokrenuta, kojom prilikom se alocira područje 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 zajednički koriste od strane procesa BP.
- Pozadinski procesi obavljaju funkcije u ime pozivajućeg procesa . Obavljaju I/O i nadziru druge Oracle procese.

Detaljna Arhitektura Oracle 8 servera



Podrucje zajedničke memorije

SGA područje sadrži podatke i upravljačku informaciju potrebnu Oracle serveru. Sastavljeno je od sledećih memorijskih struktura:

- Zajednička zaliha (share pool), memoriše skoro korišćene SQL naredbe i podatke iz rečnika
- Keš Bafera BP (Data Buffer cache)
- Kes Bafera dnevnika izmena (redo log cashe)

Zajednička zaliha SGA se sastoji od :

- Keša biblioteke naredbi, koji memoriše skoro korišćene SQL instrukcije i koji se koristi od strane aerv. procesa u toku kompilacije SQL naredbi.
- Keša rečnika podataka, koji memoriše skoro korišćene definicije u BP i koji se koristi od strane serverskih procesa u cilju razrešenja imena objekata u specificiranim SQL naredbama.

- Keš bafer baze se koristi za memorisanje skoro korišćenih podataka, Ovi podaci se citaju iz i upisuju u fajlove podataka (BP).
- Serverski proces traži potrebne blokove pod. u sklopu ovog bafera, pa tek ako ih nema, traženi blok se učitava iz baze.
- Keš bafer dnevnika izmena se koristi za praćenje promena koje u bazi posredsvom instance izvedu serverski i pozadinski procesi. Ovo se koristi u slučaju potrebe oporavka baze podatka.

Oracle procesi

- Serverski procesi , koji upravljaju zahtevima koji dolaze od konektovanih korisnickih procesa.
- Pozadinski procesi, koji obavljaju asinhoni I/O. Ovi procesi mogu biti obavezni i opcionalni. Svaka instanca mora uključiti sledeće obavezne pozad, proc.:
 - ▶ **Database writer (DBWR)**
 - ▶ **Log writer (LGWR)**
 - ▶ **Checkpoint process (CKPT)**
 - ▶ **System monitor (SMON)**
 - ▶ **Process monitor (PMON)**
- **DBWR** proces ima zadatak da modifikovane blokove podataka iz SGA bafer keša upiše u fajlove podataka na disku. Omogućava odlaganje upisivanja do pojave spec. uslova. Instanca može imati do 10 DBWR procesa.
- **LGWR** ima zadatak da prepisuje podatke iz SGA Redo log bafera u aktivni Redo log fajl na disku. Obavlja se sekvencijalno, za slučaj pojave spec. uslova. Ovaj proces potvrđuje **commit** samo posle upisa **redo** informacije na disk.
- **CKPT** proces ima zadatak da ažurira informaciju o statusu BP u upravljačkim 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** proces (SYSTEM MONITOR) ima zadatak da proverava konzistentnost BP, i u slučaju potrebe, inicira oporavak BP. SMON vrši automatski oporavak instance u slučaju njenog kvara.
- **PMON** proces ima zadatak da prati korisničke procese koji pristupaju bazi i da ih oporavlja posle eventualnog pada.
- Opcionalni pozadinski procesi
 - ▶ Archiver (ARCH)
 - ▶ Recoverer (RECO)
 - ▶ Dispatcher (Dnnn)
 - ▶ Lock(LCKO),...

Zaključak

- Izabrana arhitektura DB servera bitno utiče na performanse ukupnog sistema.
- Poznavanje detaljne arhitekture konkretnog, korišćenog DBMS-a je bitno za razumevanje rada celine c/s sistema, kao i za efikasno korišćenje njegovih svojstava i alata.

Predavanje No. 4

Upravljanje podacima – 3

Upravljanje procesima – 1

Sadržaj predavanja :

- SQL i programski interfejsi
- Tipovi IS, aplikacija i korisnika
- Upravljanje transakcijama

SQL i programski interfejsi

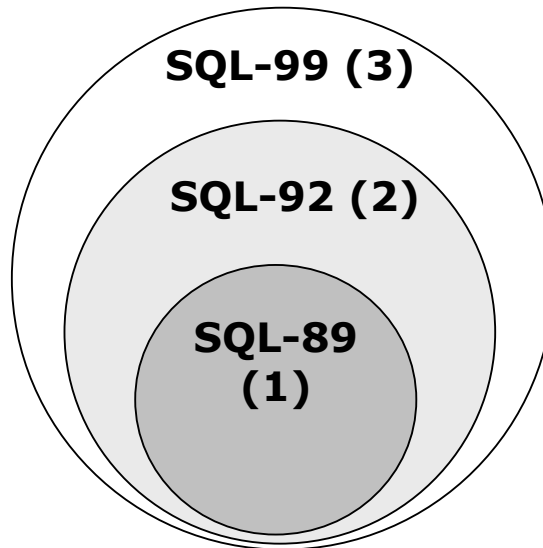
Evolucija SQL-a

Inicijativno: upitni jezik, složeno

Danas je SQL:

- interaktivni upitni jezik
- Jezik programiranja BP
- Jezik povezivanja sa umreženim BP
- Jezik definisanja i admin. Podataka

- SQL-89 (1)
- SQL-92 (2)
- SQL-99 (3)



SQL-89: osnovna svojstva

ANSI/ISO standard, presek tadašnjih implementacija, po principu min.Zajedničkog sadržaja.

- postojanje referencijalnog integriteta
- Standardizacija ugnježenog SQL-a, samo za Fortran, COBOL, PL/I, Pascal.

SQL-92: osnovna svojstva

Obim: superset 5x SQL-89

- SQL klijent-server konekcija (uključ. konkurenciju)
- Finija kontrola transakcija
- Standardizovani katalog
- Podrška ugnježenom SQL za C, Ada, MUMPS
- Podrška za dinamički SQL
- Podrška novim tipovima pod. (BLOB, VARCHAR, TIME, DATE, TIMESTAP)
- Stand.kodovi grešaka i dijagnostika
- Ograničenje i provera domena
- Podrška join operatoru
- Podrška privremenim tabelama

SQL-99: osnovna svojstva

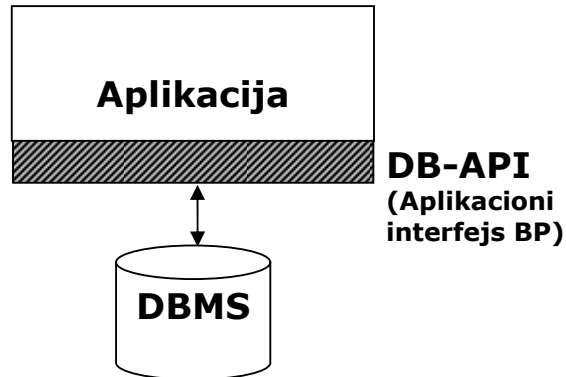
Značajno širi obim (2000 str.)

- Object SQL
- Memorisanje procedure
- Trigeri
- Korisnički definisani tipovi (UDT)
- CLI – interfejsi na nivou poziva
- Osetljivi kursori
- Multimedija
- ...

Programski interfejsi BP

Postoje dva osnovna tipa DB-API-a:

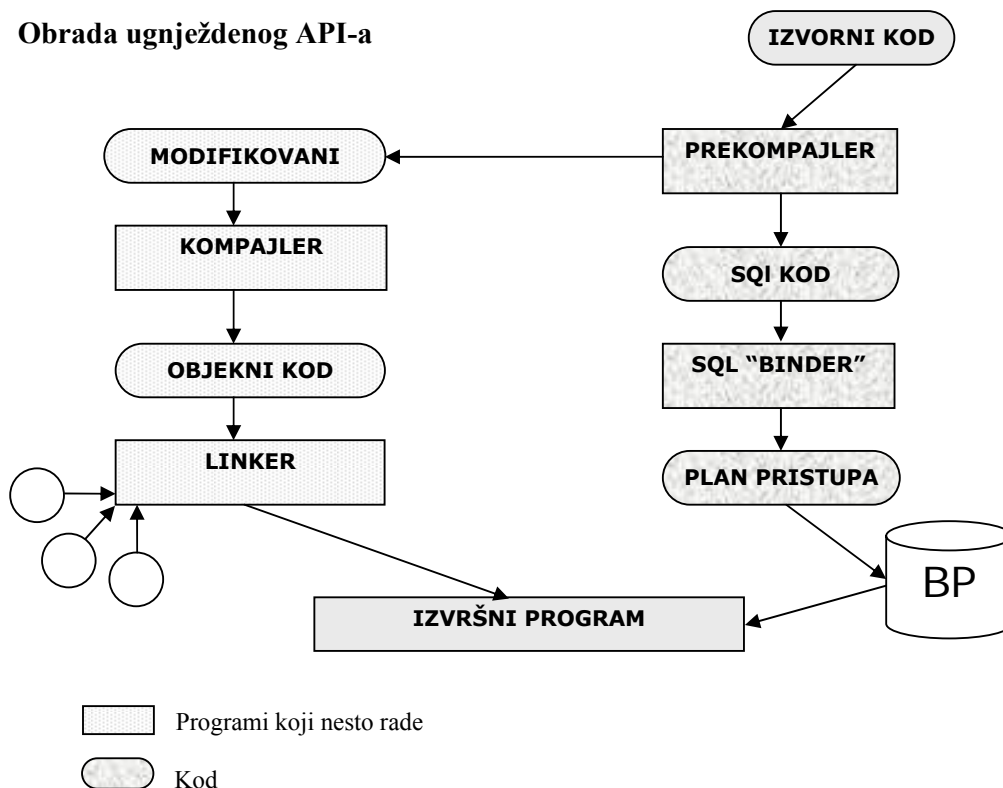
- Ugnježdeni (embedded) API (EAPI)
- API na nivou poziva (CLI-API)



Ugnježdeni API: svojstva

- SQL komande su ugnježdene u tradicion. programski jezik (2 ili 3 gen.) koji daje upravljačka svojstva (DO WHILE, GO TO).
- Prekompilacijom se SQL iskazi odvajaju od klas. Programa i odvojeno se obrađuju.
- Način pristupa BP je fiksiran i “kodiran” u programu.
- Statičke SQL instrukcije definisane u kodu i konvertovanje u plan pristupa u vreme pripreme programa (prekompilacija)

Obrada ugnjeđenog API-a

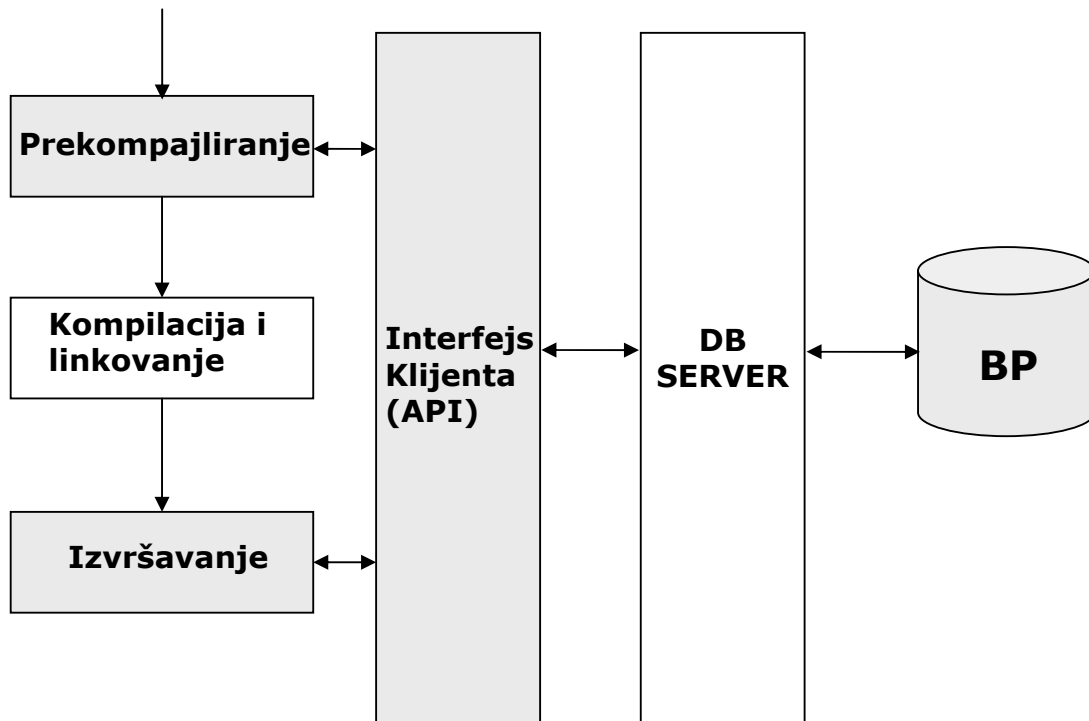


ESQL-Osnovni problemi-svojstva

- U toku razvoja aplikacije baza podataka mora biti poznata i raspoloživa
- Navedeno je veoma nezgodno za proizvođače generičkih (front-end) alata
- Mada nefleksibilan, ESQL DB-API nudi najbolje performanse.

DB-API na nivou poziva (CLI)

- Koncetualno prostiji, poziv biblioteke funkcija
- Dinamičke SQL instrukcije se kreiraju i izdaju u run-time-u, tj. U toku izvršavanja
- Tipična sekvenca:
 - Uspostavljanje veze sa BP
 - Priprema (buffera) SQL zahteva
 - Izvršavanje zahteva (obrada buffera)
 - Obrada statusa i rezultati

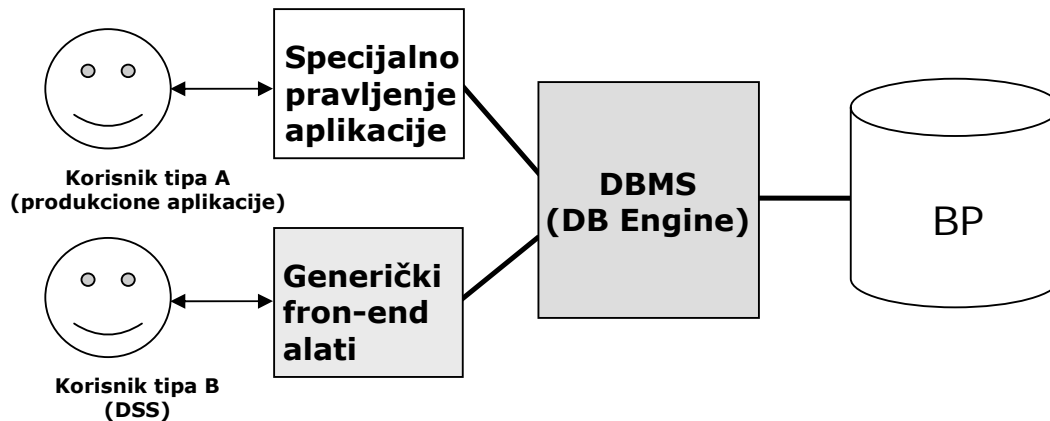
DB-API mogućnost i vreme povezivanja (CLI)

Komparacija ESQL-CLI (1-2)

SVOJSTVA	ISO SQL/CLI (CLI)	ISO SQL-92 (ESQL)
ZAHTEV DA CILJNA BP BUDE POZNATA	NE	DA
PODRŽAVA STATIČKI SQL	NE	DA
PODRŽAVA DINAMIČKI SQL	DA	DA
POTREBA ZA PREKOMPILACIJOM I POVEZIVANJEM	NE	DA
LAKOĆA PROGRAMIRANJA	NE	DA
TOOL-FRENDLY	DA	NE
LAKOĆA DEBAGOVANJA	DA	NE
LAKOĆA PAKOVANJA	DA	NE

Tipovi IS, aplikacija i korisnika

- Produkcioni sistemi: OLTP tipa (On-line transaction processing)
- Sistemi za podršku odlučivanju: DSS tipa (Decision Support System)



Produkcioni sistemi: OLTP tipa

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 odlučivanju DSS

Namena: pre svega analiza podataka

- Širi skup funkcija
- Često korišćene mat. Modela
- Manji zahtevi za pouzdanošću
- Fokus na fleksibilnost
- Primena: “Back-office”
- OLAP, DW, DM, IIS
 - OLAP - On Line Analytical Processing
 - DW - Data Warehousing
 - DM - Data Mining
 - IIS - Intelligent IS

Front-end (čeoni) alati BP: brojni

- Jednostavni alati za upit i izveštavanje (SQL*Plus)
- Alati za kompleksne analize (Oracle Express)
- Okruženja za razvoj aplikacija (Developer 2000)
Forms, Reports, Graphics,
- Generatori aplik.koda (Designer 2000)
- Alati za administriranje

Princip izbora: Jednostavni alati za jednostavne aplikacije i obrnuto.

Osnovi upravljanja procesima

- Dinamička priroda podataka
- Razmatraju se načini izvođenja kompleksnih izmena nad podacima uz održavanje njihove konsistentnosti
- Osnovu predstavlja koncept transakcije

Osnovi upravljanja procesima-

Dinamička priroda podataka: sadržaj

- Upravljanje transakcijama
- Upravljanje konkurentim pristupom
- Kontrola sigurnosti
- Problemi sa procesnom transparentnošću.

Upravljanje transakcijama

- Šta je **transakcija**?
- Svojstva **ACID**
- **Commit i Rollback**

Transakcije – Definicija (značaj)

- **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 **komitovana** (izvršena).
- Ukoliko **transakc. nije uspešna**, za trans. se kaže da je vraćena na početak (**rolled back**) ili napuštena (**aborted**)

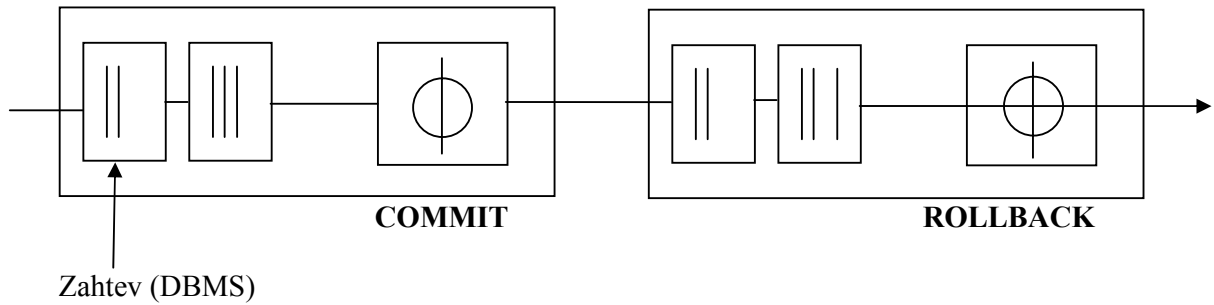
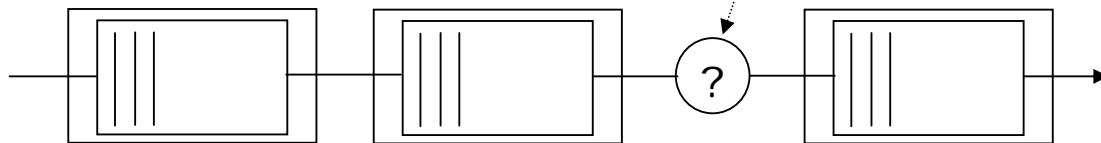
Svojstvo ACID: Atomicity, Consistency, Isolation, Durability

- **Atomarnost**: zahtev za nedeljivu jedinicu posla
- **Konsistentnost**: zahtev da aplik. Pomera BP iz jednog u drugo konsistentno stanje
- **Izolacija**: zahtev da jedna transakcija nemože uticati na ponašanje druge transakcije → zahtev za serilizacijom podataka
- **Trajnost**: zahtev da su efekti transakcije posle komitovanja, trajni.

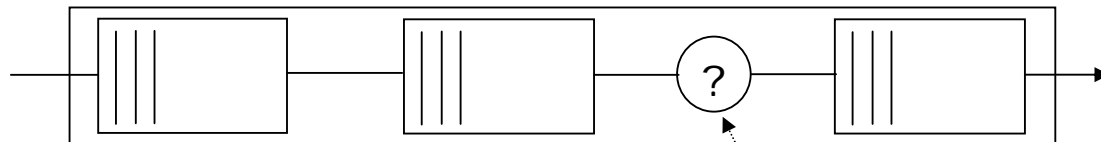
Commit i Rollback

U smislu SQL se definišane komande koje aplikaciji omogućava da specifikira:

- **BEGIN TRANSACTION**
- **COMMIT** promene u BP čini trajnim
- **ABORT/ROLLBACK** poništava izmene koje je napravila tekuća transakcija

Vizualizacija transakcije**Kratke i duge transakcije****Kratka transakcija****Akcija čoveka**

Aplikacija ima kontrolu i može da radi druge poslove

Duga transakcija**Akcija čoveka**

Transakcija traje onoliko koliko se čovek čeka.

Odnos kratkih i dugih transakcija

Tip transakcije	Efikasnost	Fleksibilnost Interakcije korisnika
Kratka (paketna)	Dobra	Slaba
Duga (konverzaciona)	Slaba	Dobra

Upravljanje konkurentnim pristupom

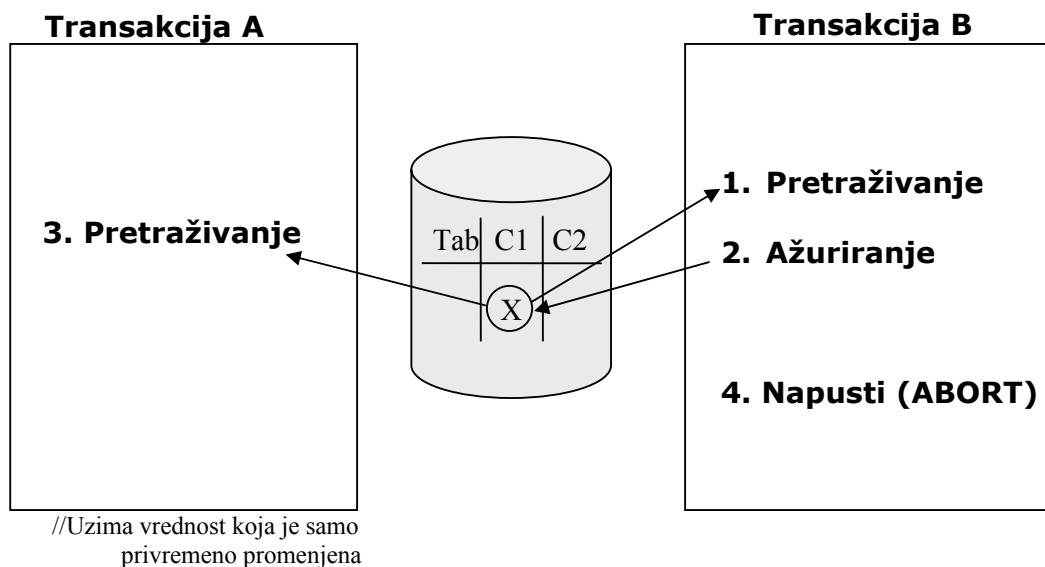
Kada se više transakcija konkurentno izvršava na BP.

- **Konkurentno:** nekoliko aktivnosti u istoj vremenskoj jedinici.
- **Simultano:** nekoliko aktivnosti u potpuno istom vremenskom momentu

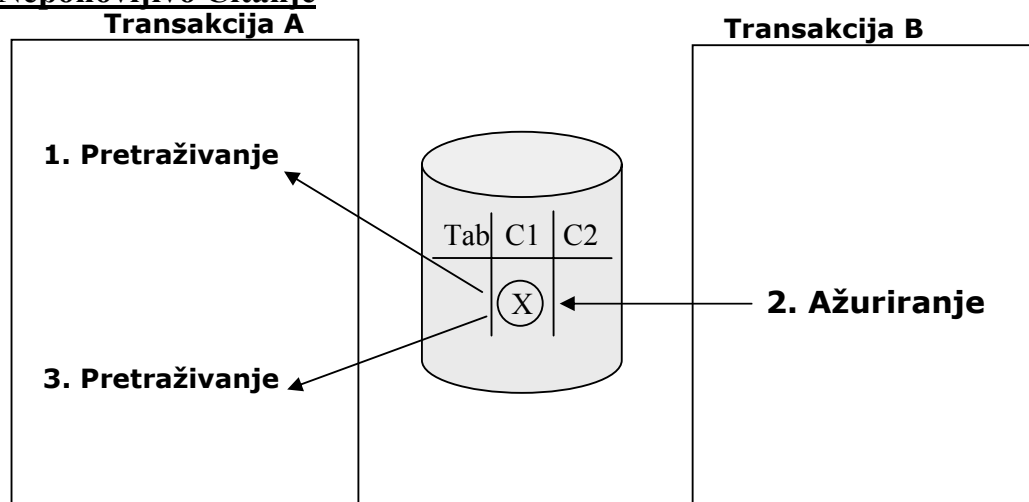
Potencijalni problemi konkurentnosti

- Nečista (prljava) čitanja
- Neponovljiva čitanja
- Fantomski redovi
- Izgubljena ažuriranja

1. Nečista (prljava) čitanja

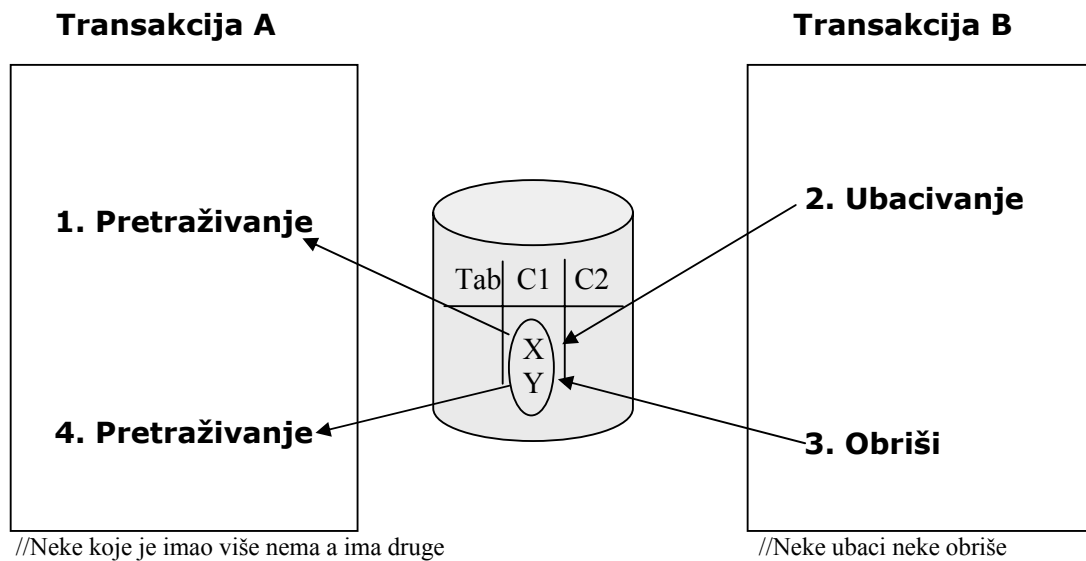


2. Neponovljivo Čitanje

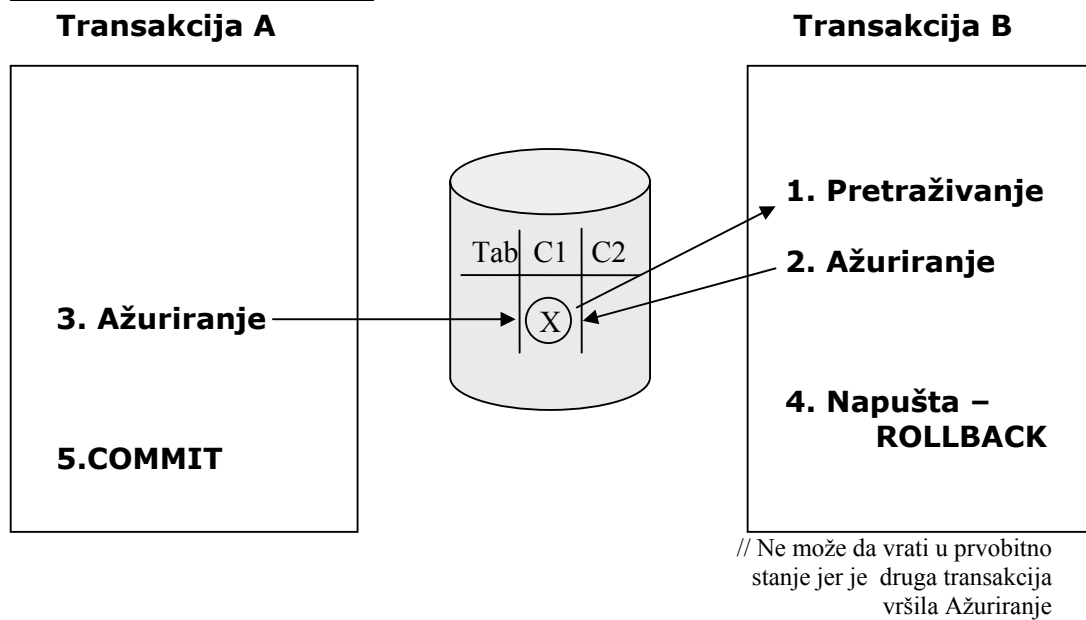


// Ista veličina u istoj tabeli ima dve različite vrednosti

3. Fantomski Redovi



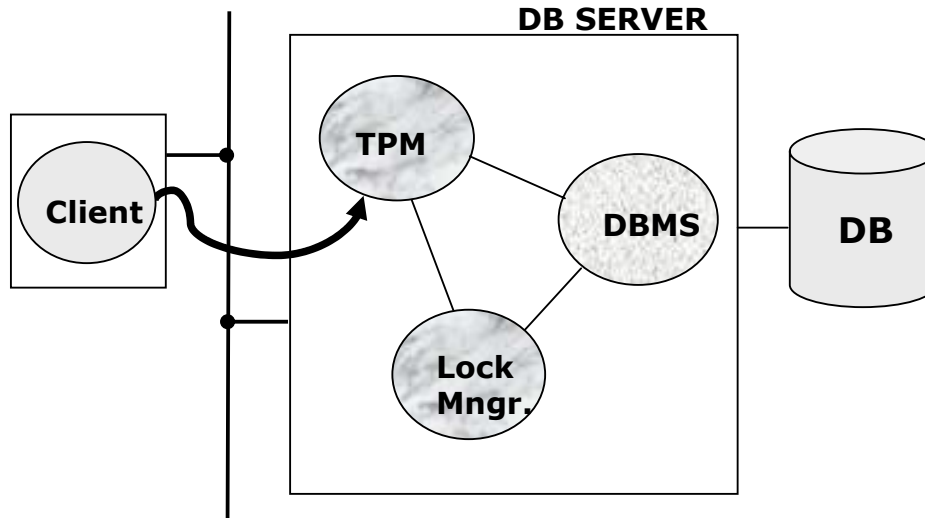
4. Izgubljena ažuriranja



Upravljanje konkurentnim pristupom -pesimistički pristup

- Pesimistički-tradicionalno:
zaključavanjem objekata BP pre njihovog korišćenja.
- Na referentni objekat BP se postavlja **lock** (katanac)
- Tehnika je efikasna ali posledica na performanse
- Moguća pojava **dead-lock**-ova.

Upravljanje transakcijama primenom TPM na serveru



Upravljanje konkurentnim pristupom – pesimistički pristup Moguća poboljšanja

- Tipovi zaključavanja
- Granularnost zaključavanja
- Nivoi izolacije
- Trajanje zaključavanja
- Detekcija i razrešavanje **dead-lock**-ova

Tipovi, Nivoi lock-ova

Tipovi lock-ova:

- Deljenji
- Ekskluzivni

Nivoi zaključavanja:

- Nivo reda
- Nivo stranice
- Nivo tabele

Obuhvat i trajanje zaključavanja

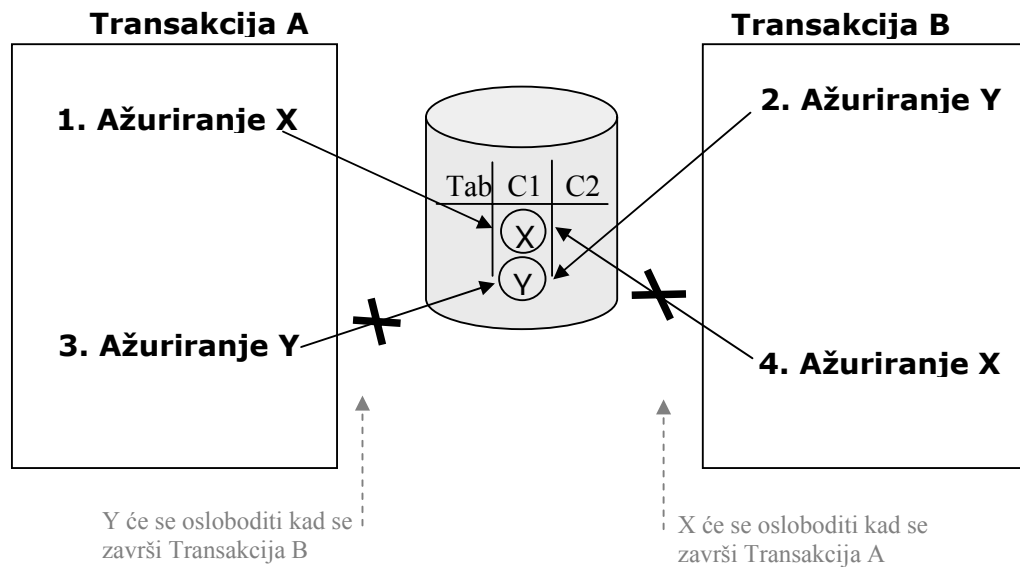
Se može menjati tokom izvršenja transakcije.

- Potpuna izolacija (ponovljivi “read”)
- Delimična izolacija (stabilnost kursora)

Trajanje lock-a:

Tipično se lock-ovi prikupljaju kada se obj. referencira a otpuštaju kada se transakcija završi.

Detekcija i razrešavanje dead-lock-ova



Jedna drugu sprečavaju da se završe –Dead-Lock.
Sistem se brani tako što jednu transakciju odbaci, da bi se druga završila.

Alternativa metodu zaključavanja

Pristup sa **vremenskim markiranjem** početka transakcija, čime se lock-ovi izbegavaju, a kod konflikta dolazi do rollback-a i restarta

Optimistički pristup kontroli konkurentnosti, gde se validacija pod. vrši neposredno pre commita, tj. vrši se provera da li je dolazilo do konflikta.

Na bazi pretpostavke da je nivo rizika nizak.

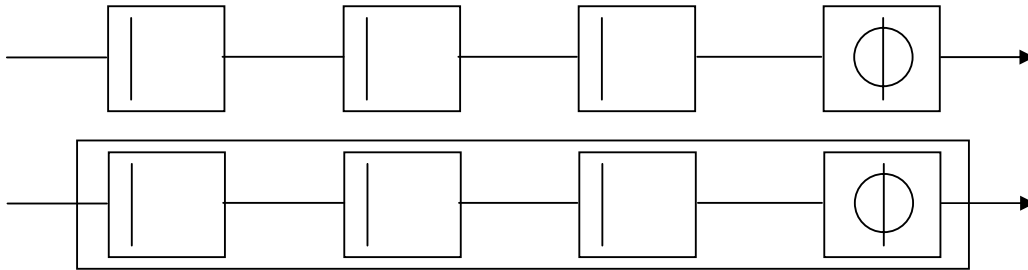
Upravljanje sigurnošću

- DB privilegije: GRANT/REVOKE
- Definicija pogled-a
- Memorisanje procedure

Problemi sa transparentnošću procesa

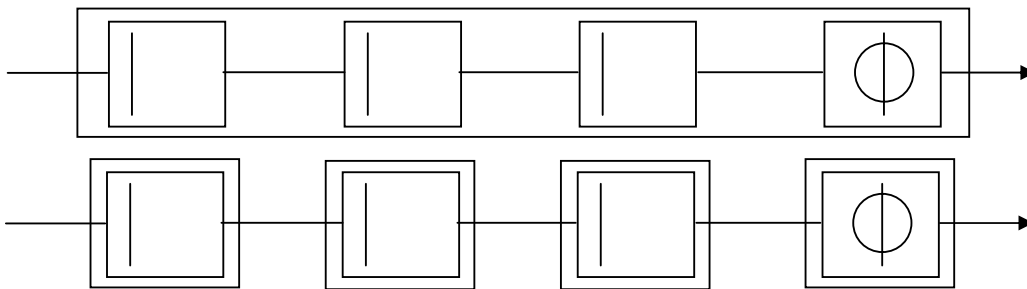
Posledica nesklada dva okruženja u kojima se izvršava ista transakcija u pogledu:

- 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 početak transakcije, nego samo jedan korak unazad.

U drugačijem okruženju (druga slika), transakcija će se drugačije ponašati, i usled neuspeha nekog koraka vraća na početak cele transakcije. Slika prikazuje jedni te istu transakciju na dva različita okruženja.



// Situacija simetrična prethodnoj. Transakcija je inicijalno projektovana bez Auto Commit-a posle svakog koraka, ali kada radi u drugačijem okruženju, drugačije se ponaša, tj vršiće se Auto-Commit posle svakog koraka.

Distribuirani sistemi

- **Osnovne definicije:**
 - ▶ **Distribuirana obrada**
 - ▶ **Distribuirana baza podataka**
 - ▶ **Multi database sistem (MDBS)**
 - ▶ **Distribuirani DBMS(DDMBS)**
- **Načini distribucije podataka**
- **Načini distribucije procesa/transakcija**

Distribuirana obrada:

Izvršavanje kooperativnih procesa koji međusobno komuniciraju putem razmene poruka preko (informacione) mreže.

Realizuje se kao:

Centralizovana baza podataka kojoj se se može pristupiti preko računarske mreže.

Distribuirana baza podataka:

Logički integrisani skup deljenih (zajedničkih korišćenih) podataka koji su fizički distribuirani na više servera koji su povezani (informacionom) mrežom.

Multi database sistem, tj. Sistem sa više baza:

Distribuirani DBMS u kome svaka lokacija održava kompletnu autonomiju.

Federalizovani database sistem, tj. Sistem federalizovanih baza podataka?

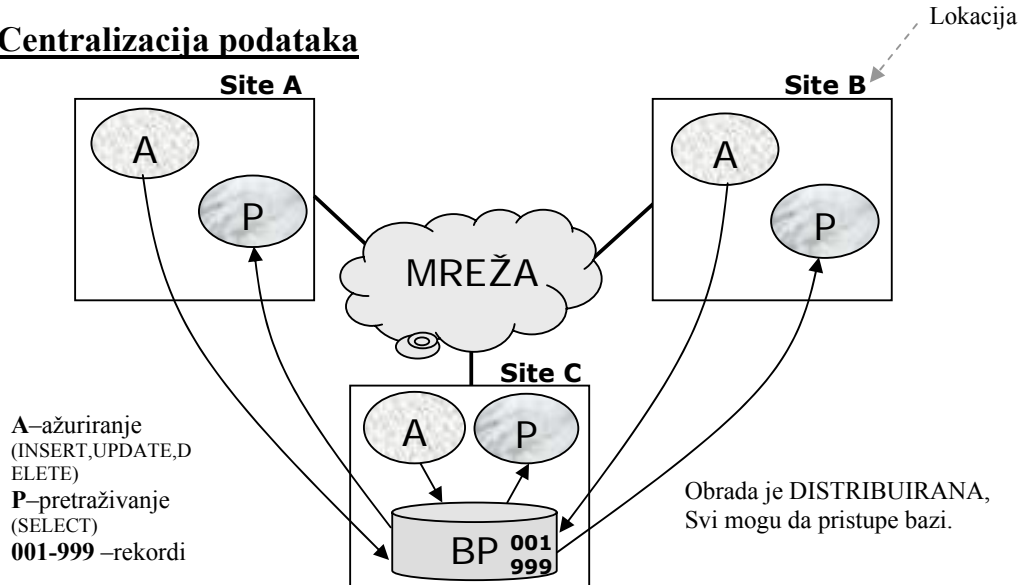
Distribuirani DBMS (DDBMS):

se definiše kao s/w za upravljanje distribuiranom bazom podataka, na takav način da su aspekti distribucije transparentne za korisnika.

DDBMS podržava razvoj DBP koje korisniku izgledaju i ponašaju se kao nedistribuirane BP.

Centralizacija ili distribucija?

- Kada centralizovati podatke ?
- Kako i kada distribuirati podatke?

Centralizacija podataka**Centralizacija podataka podesna:**

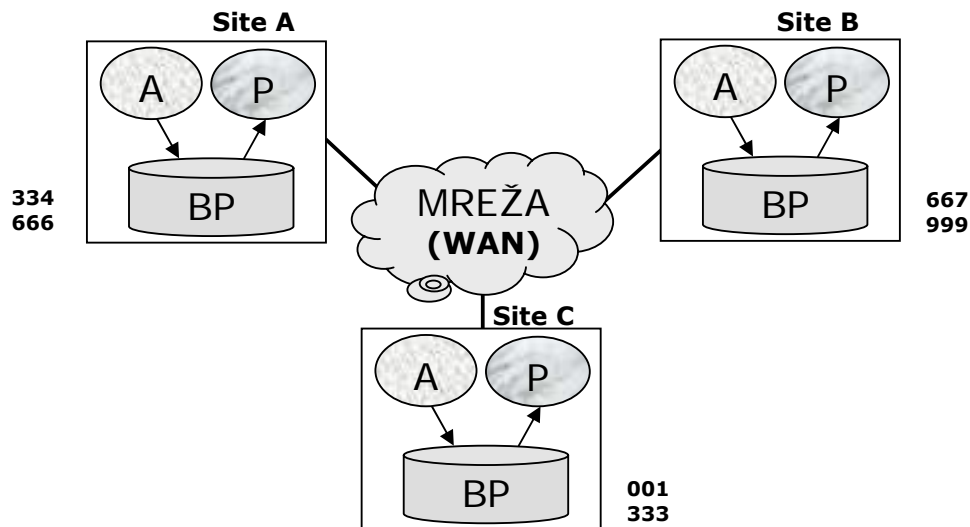
- Kada se veći broj podataka često ažurira iz više lokacija.
- Kada se podaci u BP veoma često menjaju.
- Kada klijent želi da obavi posao iz bilo koje lokacije (poslovnica)
- Primeri: sistemi za rezervaciju karata, real-time upravljački sistemi.

Načini distribucije podataka:

- Partitioniranje
- Ekstrakcija
- Replikacija
- Keširanje

Partitioniranje: horizontalno ili vertikalno (fragmentacija)

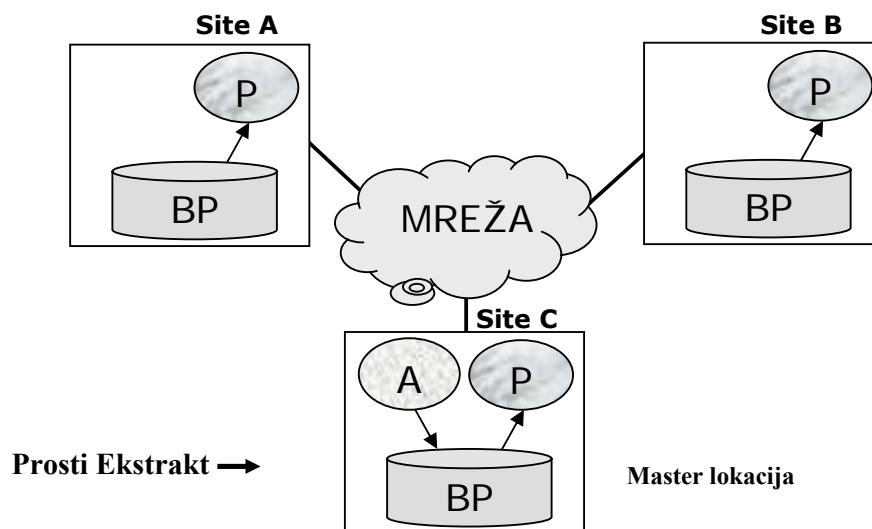
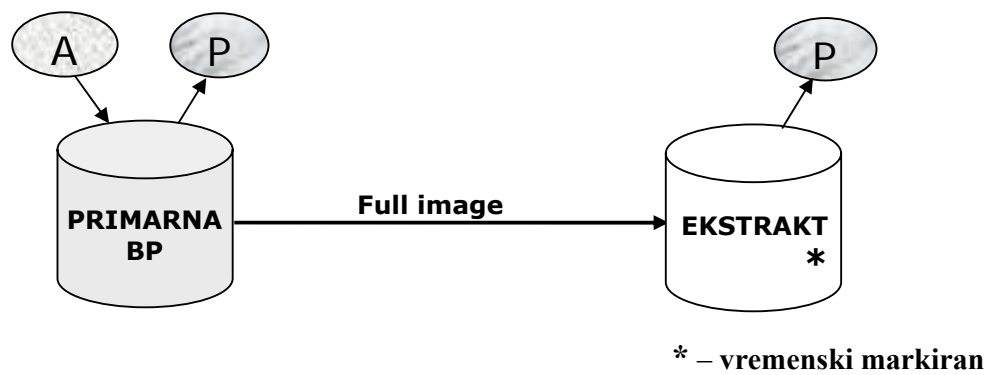
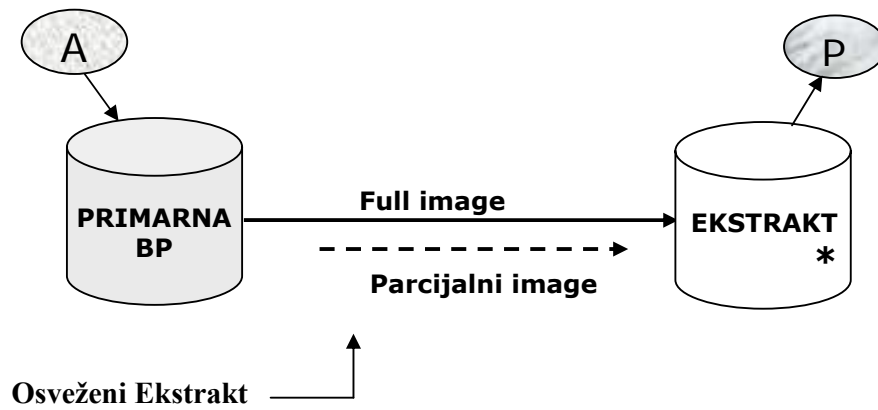
- Tabela se deli po horizontali (podskup redova) ili po vertikali (po atributima) manje delove (fragmente) bez preklapanja, koji se distribuiraju na 2 ili više servera.
- Prednosti: Korišćenje (pogledi), efikasnost, paralelizam, sigurnost.
- Nedostaci: performansa, integritet
- Primeri: ED preduzeća, osiguranje...

Partitioniranje: horizontalno**Ekstrakt:**

- Prosta i jeftina tehnika distribucije podataka zasnovana na kopiranju
- Primarna baza i njena kopija, obično samo deo (ekstrakt).
- Ima smisla ako su podaci nepromenljivi ili sporo promenljivi.
- OK za istorijske, arhivske podatke.
- Ekstrakt se **ne ažurira**.
- Opasnost pojave ekstrakcionog haosa

Postoji razni tipovi ekstrakta:

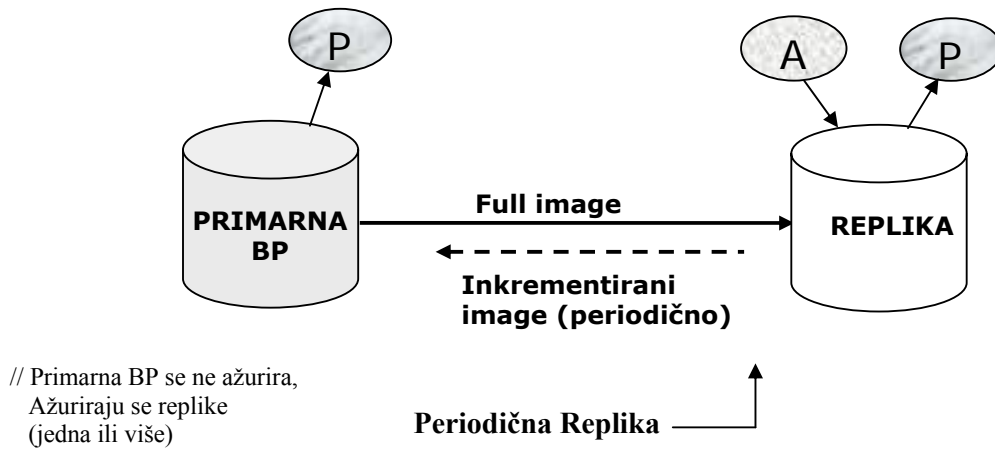
- Prosti ekstrakt
- Vremenski markirani ekstrakt
- Osveženi ekstrakt

Prosti ekstraktVremenski markirani ekstraktOsveženi ekstrakt

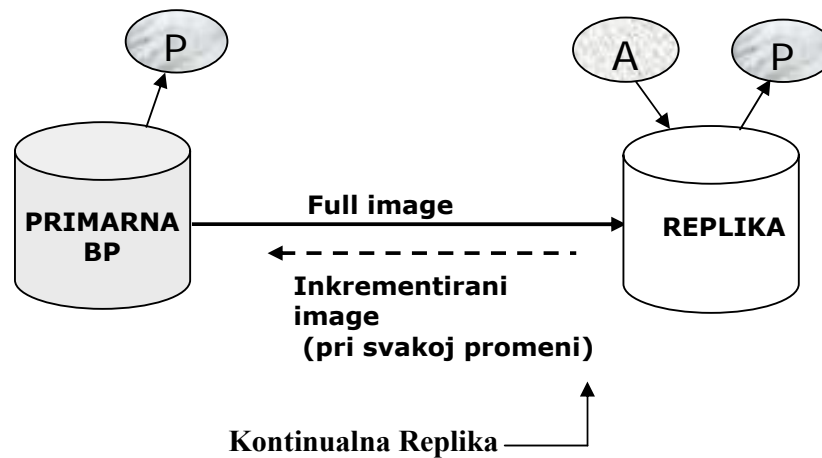
Replika

- Kopija **master BP** koja se **može ažurirati**.
- Periodično se vrši sinhronizacija master BP slanjem **inkrementiranog imidža** sa replike.
- Danas veoma često korišćena tehnika distribucije podataka.
- Značaj adekvatne kom.mreže.

Periodična replika



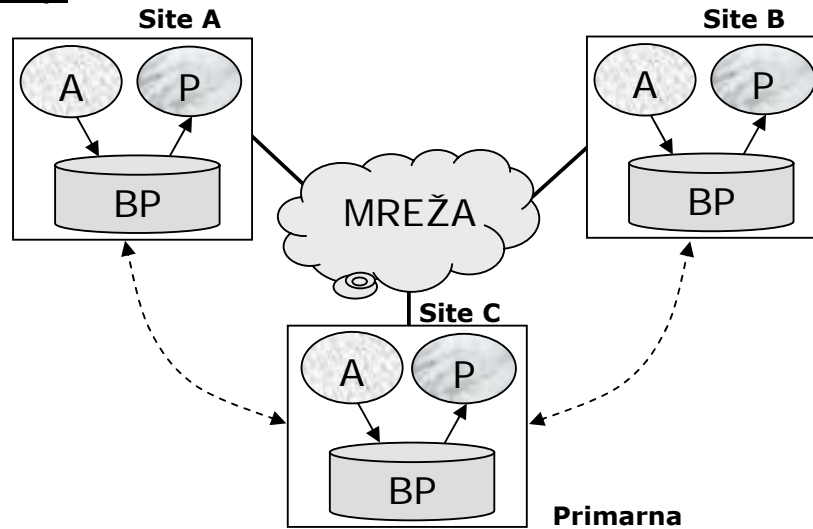
Kontinualna replika



Keširanje

- Predstavlja tehniku distribucije podataka gde se pravi privremena (dinamička) replika BP ili njenog dela
- Može biti na serveru ili na klijentu
- Poboljšava performanse rada ako:
 - ▶ Se jednom referisani podaci (grupa) potom koriste (npr. Internet)
 - ▶ Se podaci retko menjaju
- Problem: mehanizam održavanja konsistentnosti keševa.

Keširanje



Načini distribucije procesa

- Moguće različite klasifikacije transakcija
- Ovde IBM DRDA (Distributed Relational Database Architecture) bazirana.
- Četiri tipa transakcija, sa rastućim nivoom kompleksnosti

Zahtev = SQL instrukcija

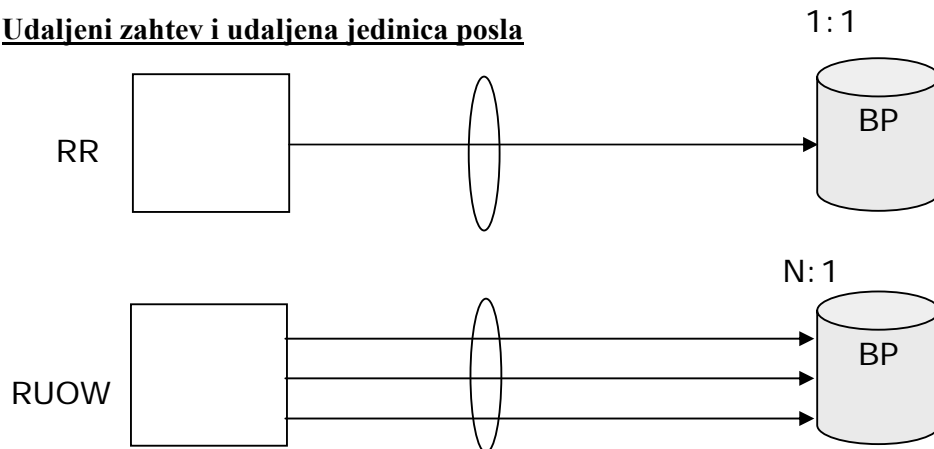
UOW = transakcija

// UOW–unit of work

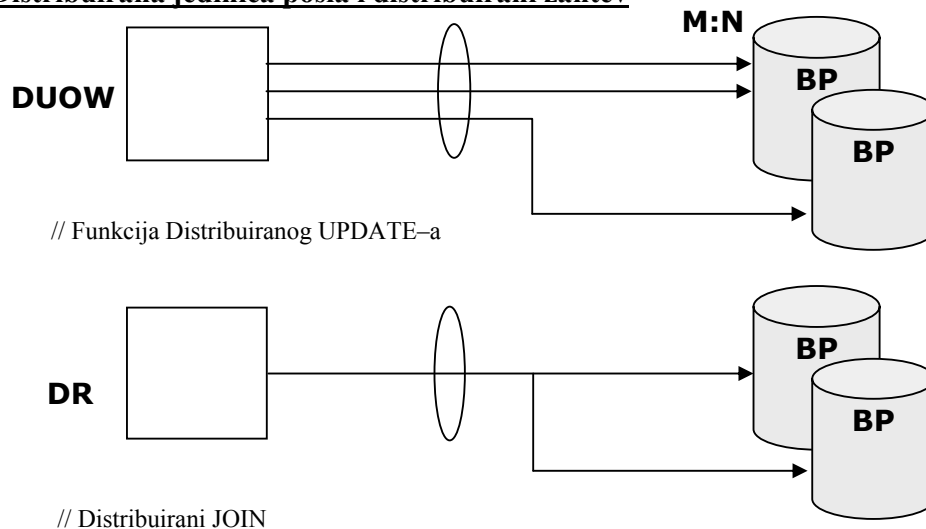
Načini distribucije procesa:

- Udaljeni zahtev (**Remote Request**) –RR
- Udaljena jed. Posla (**Remote UOW**) –RUOW
- Distribuirana jed. Posla (**Distrib.UOW**) –DUOW
- Distribuirani zahtev (**Distrib.Request**) –DR

Udaljeni zahtev i udaljena jedinica posla



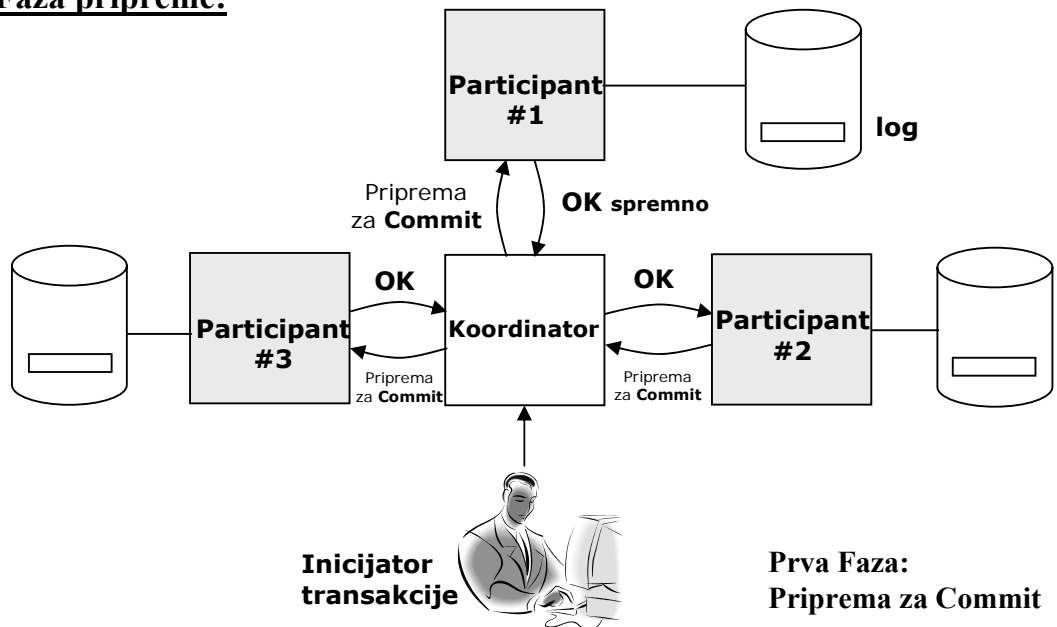
Distribuirana jedinica posla i distribuirani zahtev



Protokol dvofaznog izvršavanja (2PC):

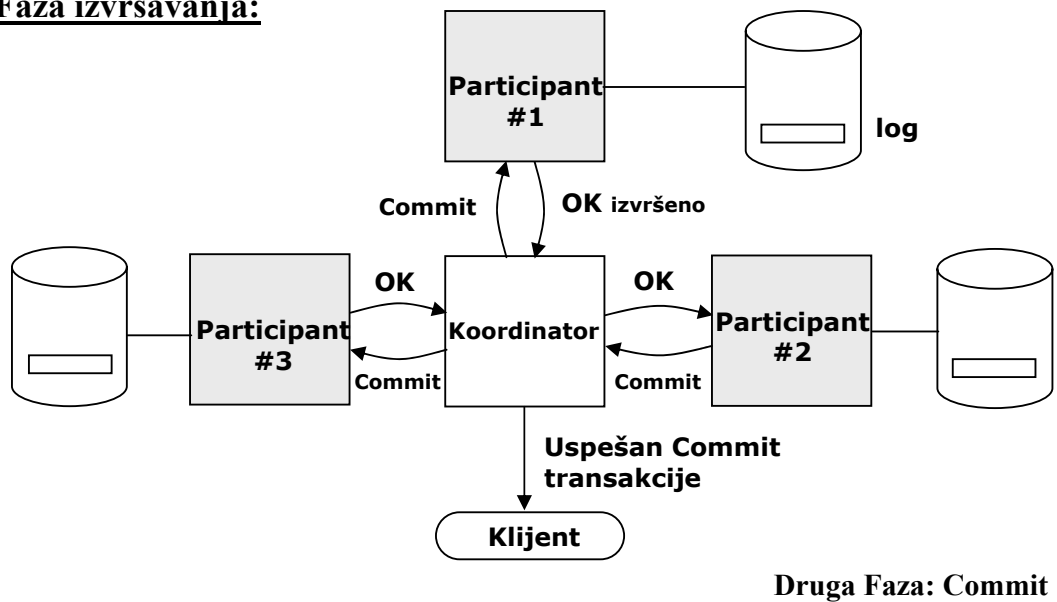
/2PC – Two Phase

Commit

Faza pripreme:

// Korisnik inicira transakciju. Svaka mašina (Participant, DBMS) je nadležna za akcije koje se njega tiču. Koordinator daje glavni zahtev za Commit.

Kad svi odgovore ide se na drugu fazu: **Globalni Commit**

Faza izvršavanja:

Distribuirani DBMS (DDBMS)**C. Date-ovih 12 pravila (iz 1987.) za DDBMS:**

1. Lokalna autonomija.
2. Bez oslanjanja na centralnu lokaciju.
3. Kontinualni rad.
4. Nezavisnost od lokacije.
5. Fragmentaciona nezavisnost.
6. Replikaciona nezavisnost.
7. Obrada distribuiranih upita.
8. Obrada distribuiranih transakcija
9. Hardverska nezavisnost
10. Nezavisnost od OS.
11. Nezavisnost od mreže.
12. Nezavisnost od DBMS.

Poslenja 4 pravila su naravno ideal.

Distribuirani sistemi

- Multi database (MDBMS) sistemi
- Federalizovani MDBMS sistemi

Osnovne definicije

Multi database sistem, tj. Sistem sa više baza:

Distribuirani DBMS u kome svaka lokacija održava kompletnu autonomiju.

Federalizovani database sistem, tj. Sistem federalizovanih baza podataka.

Trenitak ažuriranja replika:

- **Sinhrona replikacija:** Replicirani podaci se ažuriraju odmah posle ažuriranja izvorne BP, korišćenjem 2PC. (npr. Kod finansijskih transakcija)
- **Asinhrona replikacija:** Ciljna Baza se ažurira posle modifikacije izvorne baze. Zadržka može biti od $n \cdot \text{sec}$. Do nekoliko sati ili dana. Kompromis raspoloživosti i integriteta podataka.

Vlasništvo nad podacima

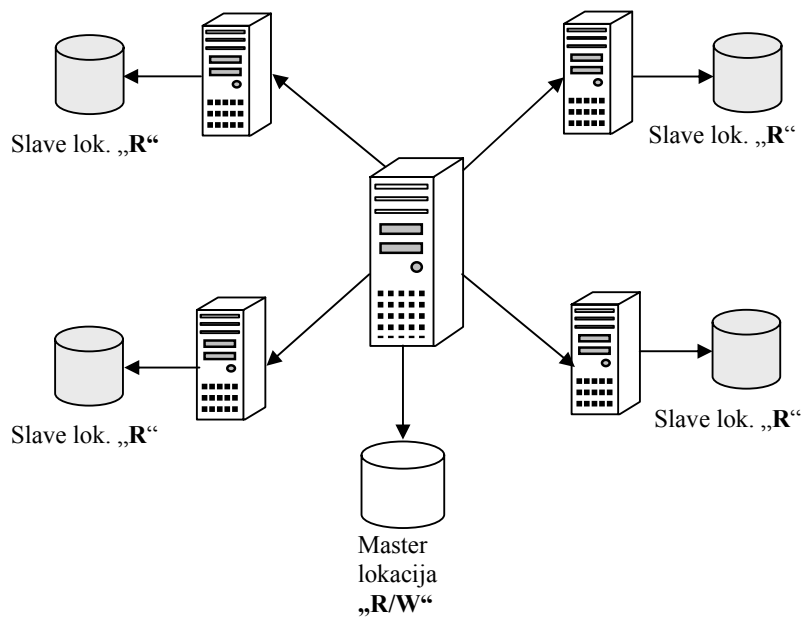
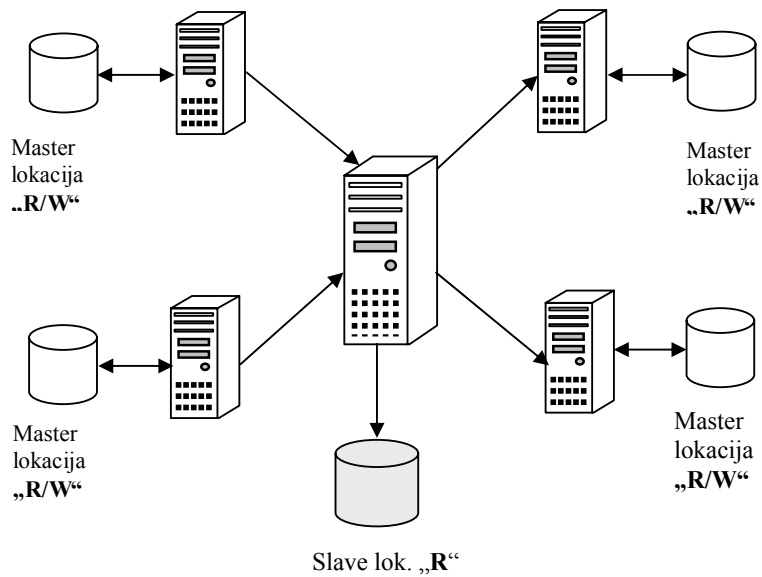
1. „**Master / Slave**“ tipa:
 - Disiminacija podataka
 - Konsolidacija podataka
2. „**Workflow**“ tipa (pravo ažuriranja repliciranih pod. Se prenosi sa sajta na sajt)
3. „**Update anywhere**“ tipa (simetrična replikacija, P2P) gde svi sajtovi imaju ista prava u pogledu ažuriranja repliciranih podataka. Potrebna metodologija za detekciju i razrešavanje konflikata.

„Master/Slave“ tipa:

- Asinhrono replicirani podaci kod ovog tipa su vlasništvo jedne lokacije, i mogu biti ažurirani samo od strane tog site-a (lokacije).
 - Koristi se „**publish –and-subscribe**“ metafora, gde je master site izdavač koji čini pod. raspoloživim.
 - Ostali sajtovi su samo pretplatnici koji dobijaju „read only“ kopije.
- Potencijalno svaki sajt može biti master za podatke koji se ne preklapaju, ali uvek samo jedan sajt može vršiti ažuriranje master kopije određenog skupa podataka, tzv. Single-site-update.

Primeri:

- Distribucije i disiminacije centralizovane informacije
- Konsolidacije udaljene informacije

Master/Slave tipa –disiminacija**Master/Slave tipa –konsolidacija**

Distribuirani sistemi 2

- Modeli i arhitekture
- Prednosti i nedostaci DIS
- Arhitekture Distribuiranih IS

Modeli obrade podataka:

- Model centralizovane obrade
- Model personalne obrade
- Model distribuirane obrade

Navedene modele obrade podržavaju odgovarajuće arhitekture

Terminologija:

Arhitektura (Webster): umetnost i nauka projektovanja i gradnje objekata

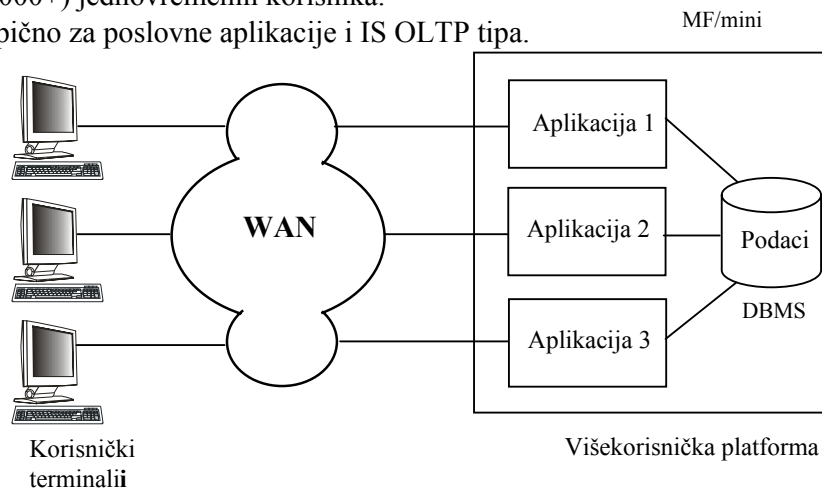
Arhitektura IS: pristup projektovanju strukture IS, tj. Definisanju njegovih osnovnih elemenata (podсистема) i njihovih međuveza.

Arhitektura informacionih sistema

- Centralizovana, više korisnička
- Distribuirana, jedno-korisnička
- Distribuirana, više-korisnička

Centralizovana, više korisnička arhitektura

- Ova arh. se realizuje putem mreže terminala priključenih na centralni (host) računar većeg kapaciteta (MF).
- Praktično sve komponente sistema (funkcije, podaci) se nalaze i izvršavaju na centralizovanoj računarskoj platformi i koriste od strane većeg broja (200-10000+) jednovremenih korisnika.
- Tipično za poslovne aplikacije i IS OLTP tipa.



Prednosti i nedostaci centralizovane više korisničke arhitekture

Prednosti:

Tehnologija stabilna, pouzdana, dobro podržana. Obezbeđuje funkcije i pristup podacima za > 1000 korisnika.

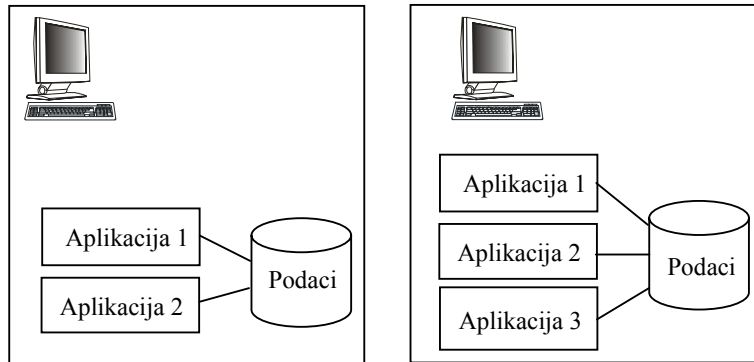
Nedostaci:

Tehnologije proizvođačke, međusobno nekompatibilne, skupe za nabavku i implementaciju, značajni troškovi dogradnje.

Distribuirana jedno-korisnička arhitektura

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

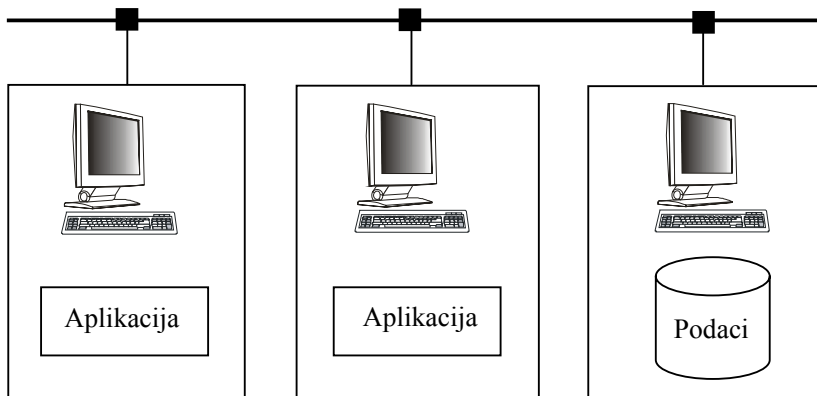
Distribuirana jedno-korisnička arhitektura (izolovana)



Distribuirana više korisnička arhitektura

- Ova arhitektura se realizuje sa više računara, njihovim povezivanjem u lokalnu računarsku mrežu (LAN).
- Komponente sistema (funkcije, podaci) se mogu nalaziti na različitim računarima, obično su podaci na jednom računaru koji ima funkciju file-servera.

Distribuirana više korisnička arhitektura sa fajl-serverom



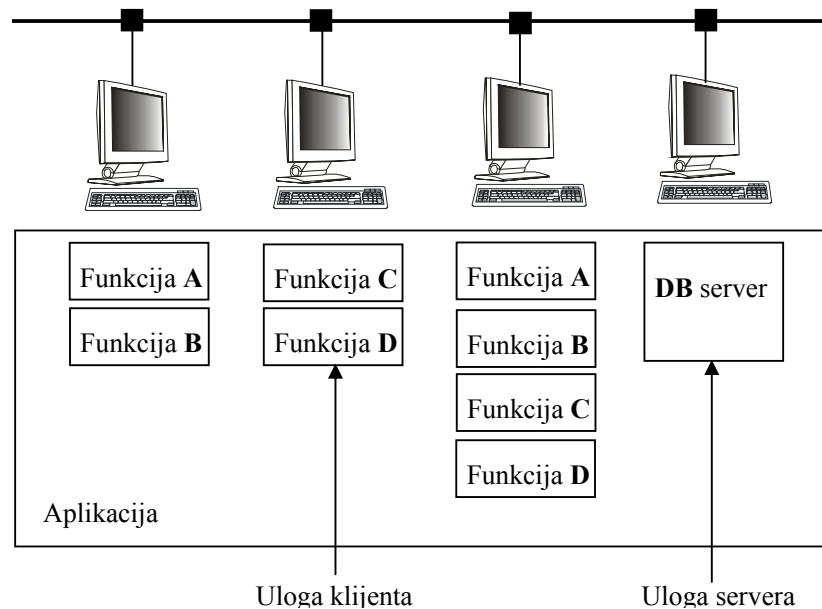
Distribuirana više korisnička arhitektura sa fajl serverom

Nedostaci:

- Intezivan saobraćaj 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,...)

Distribuirana više korisnička arhitektura klijent/server tipa**Prednosti distribuiranih IS**

- Poboljšana fleksibilnost
- Lokalna autonomija
- Povećana pouzdanost i raspoloživost (kroz redundansu i fault tolerance).
- Poboljšane performanse
- Lokalizacija proboja sigurnosti.
- Računari 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

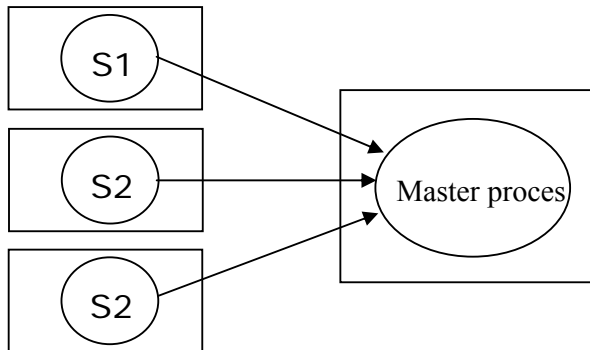
- Teže ih je upravljati, zato što su znatno **kopleksniji**, u domenu administracije, održavanja, obezbeđenja sigurnosti.
- Mnogo više komponenti koje potencijalno mogu otkazati. Prosečne komponente (PC) na početnu bilo znatno manje pouzdane od MF sistema.
- Na početku, nedostatak iskusnih osoblja za podršku i razvoj, slabija podrška isporučioaca. Potreba za **integratorima sistema**.

Arhitektura DIS

Postoji šest osnovnih paradigmi za struktuiranje distribuiranih IS:

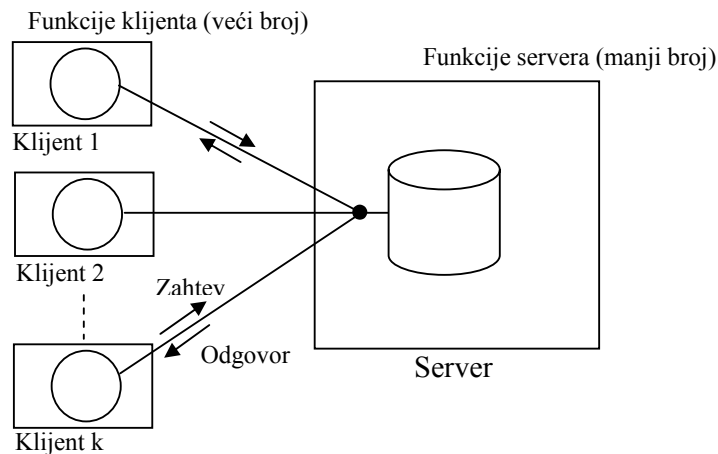
- “Master/Slave”
- Klijent–server
- Model od sloja do sloja //P2P peer-to-peer
- Grupni model
- Model distribuiranih objekata
- Model multimedijalnog toka

“Master-slave” model

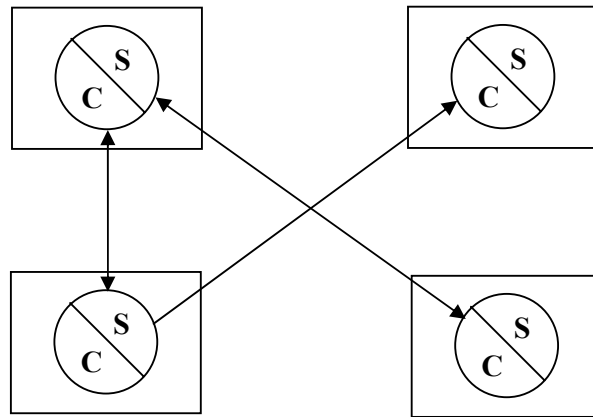


- Master proces inicira i kontroliše svaki dijalog sa drugim (slave) procesima.
- Slave proces odgovarajući na komande sa mastera, samo kada je pozvan.
- Master proces def. Skup komandi i odgovarajuće odgovore.
- Ovo je bio model na kome su se bazirali on-line centralizovani sistemi (time-sharing IS).

Klijent-server arhitektura



- Ovaj model (arhitektura) je najšire korišćena paradigma za struktuiranje distribuiranih sistema.
- 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 veći broj interfejsa.
- Interakcija klijenta i servera se odigrava po principu zahtev/odgovor, korišćenjem odgovarajućeg protokola tipa zahtev/odgovor.
- Klijentski i serverski procesi su ravnopravni
- Korišćenje manjeg broja servera poboljšava upravljivost.

Model od sloja do sloja (P2P)

- Ova arhitektura eliminiše potrebu za serverima, omogućava pojedinačnim računarima da dele resurse (aplikacije, diskove, procesor) i da međusobno komuniciraju na istom nivou.
- Svaki učesnik ima jednake mogućnosti i odgovornosti.
- P2P obično kreću sa deljenjem fajlova i resursa (npr. Printeri).

Primer: P2P okruženje “Napster” omogućava zajednici korisnika lako deljenje muzičkih fajlova. ICQ trenutni messaging servis.

Prednosti: Mogućnost da se upotrebi neiskorišćeno proc. Snaga umreženih računara. (SETI proj. n*10.000 rač.), mogućnost uspostavljanja WG sa pristupom svačijim fajlovima bez intervencije sa centralnog mesta.

Nedostaci: P2P mreže su skoro uvek jednostavnije i jeftinije od C/S ali otvaraju brojna pitanja u pogledu:

- Performansi, mogu biti lošije pri velikom opterećenju
- Sigurnosti mreže, pri odsustvu tradicionalne administracije i zaštite.

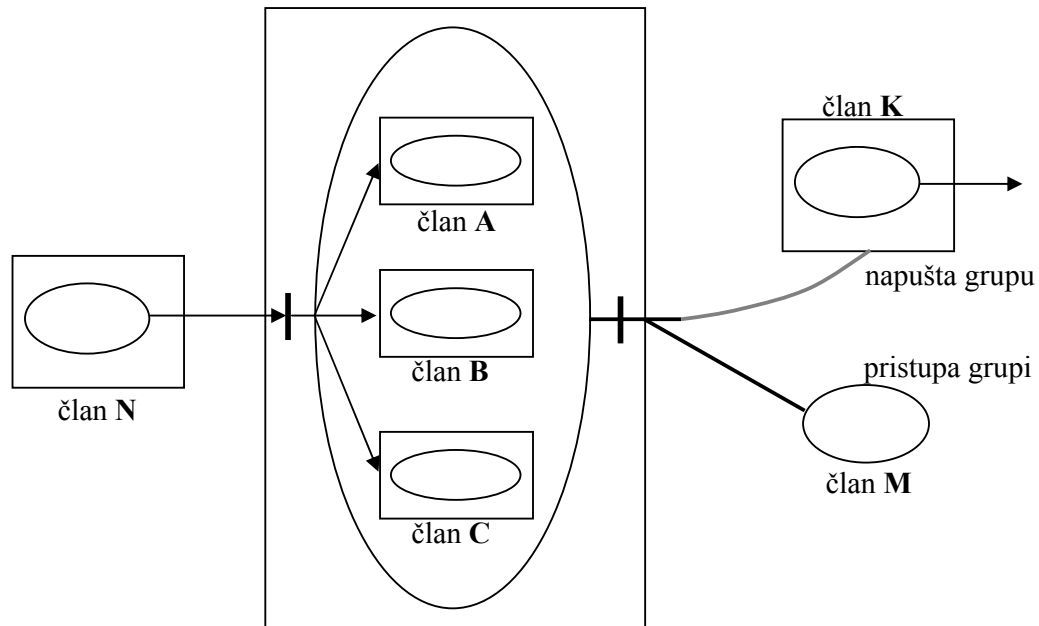
Grupni model (Groupware)

- GW je softver koji podržava kreiranje, tok i praćenje nestruktuirane informacije a kao direktna podrška kolaborativnoj grupnoj aktivnosti.
- GW je uključen u upravljanje i informacijama i aktivnostima.
- Razlika u odnosu na DBMS:
 - GW se bavi visoko nestruktuiranim pod.
 - Organizuje ih u dokumente, bazična jedinica upr.
 - GW pomera dok. Putem e-pošte i DB replika
 - Kreira baze podataka dokumenata.
- Skup tehnologija koje omogućavaju predstavljanje kompleksnih procesa koji su centrirani oko koleborativnih ljudskih aktivnosti. 5 baznih tehnologija:
 - Upravljanje multimedijalnim dkumentima
 - Workflow (tok posla, odvijanje posla)
 - E-mail
 - Konferencije
 - Planiranje (scheduling)

- Ni jedan GW proizvod za sada ne obuhvata sve tehnologije

Primeri:

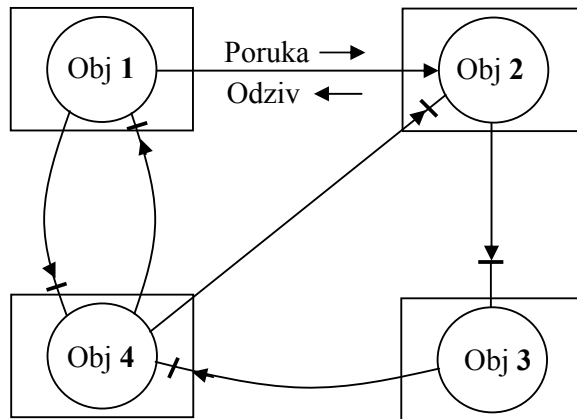
- Lotus Notes/Domino 5.0
- Novell GroupWise (nad Netware 5.0)
- MS Exchange



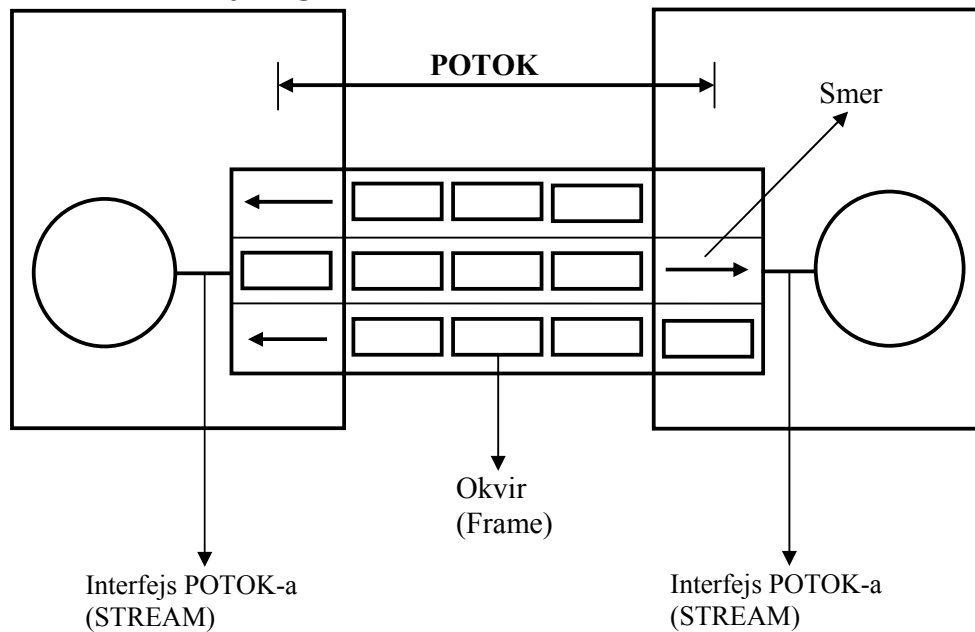
- Koristi se u slučajevima gde grupa procesa treba da sarađuje na taj način što jedan proces treba da pošalje poruku svim ostalim procesima u grupi i dobije odgovor od 1 ili više članova
- npr. video konferencije
- Kada se poruka šalje interfejsu grupe dobiće je svi članovi grupe.

Osnovni pristupi rutiranja poruka do svakog člana grupe:

- **Unicasting**, slanje posebne kopije poruke, individualno rutiranje do svakog člana grupe. Mora znati ind. adresu.
- **Multicasting**, slanje jedne poruke na adresu grupe, koja se koristi za potrebe rutiranja. Ovo je efikasan meh. pošto je broj mrežnih transakcija značajno manji nego kod Unicasting-a.
- **Broadcast**, slanje jedne poruke svim entitetima na mreži, podesno ako adrese članova grupe nisu poznate. Većina mrežnih protokola implementira ovu funkciju.

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 način se bez poznavanja načina implementacije obj. 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

Model multimedijalnog toka

- MM tok se može opisati kao kontinualni medijum sa dobro definisanim početkom i krajem koji se odigrava sa definisanom brzinom i pokazuje ne struktuirano ponašanje.
- Tok može biti označen skupom događaja koji mogu označavati promene u prezentacionom stanju ili korišćeni za okidanje akcija prikaza.

Generalizovana arhitektura MM toka, uključuje sledeće koncepte:

- Interfejs MM toka, samo jedan, onaj u kome su sve interakcije tokovi (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 “proizvođača”, čitaju od strane “potrošača”.

Distribuirani sistemi 3

Osnovne definicije

Multi Database sistem, tj. sistem sa više baza:

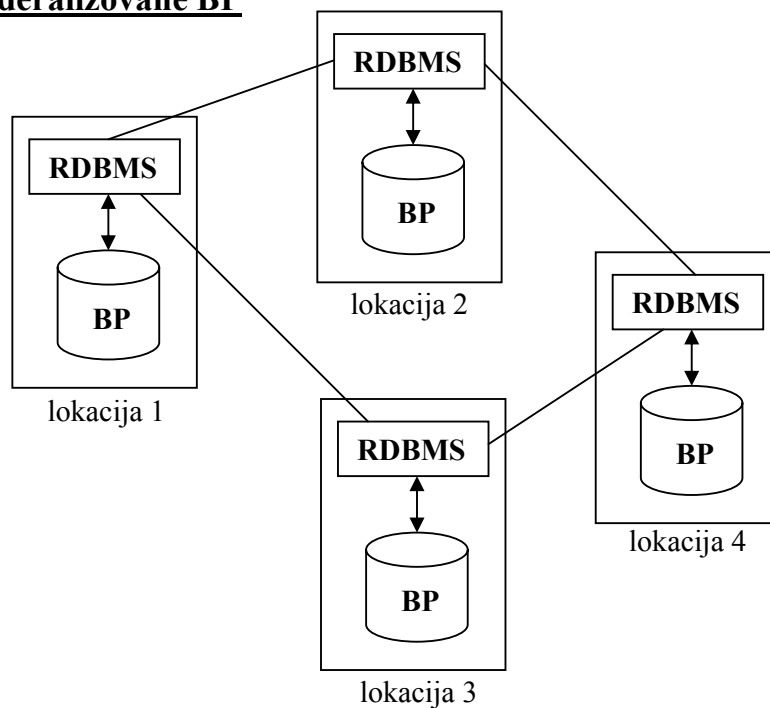
Distribuirani DBMS u kome svaka lokacija održava kompletnu autonomiju.

Federalizovani database sistem, tj. sistem federalizovanih baza podataka.

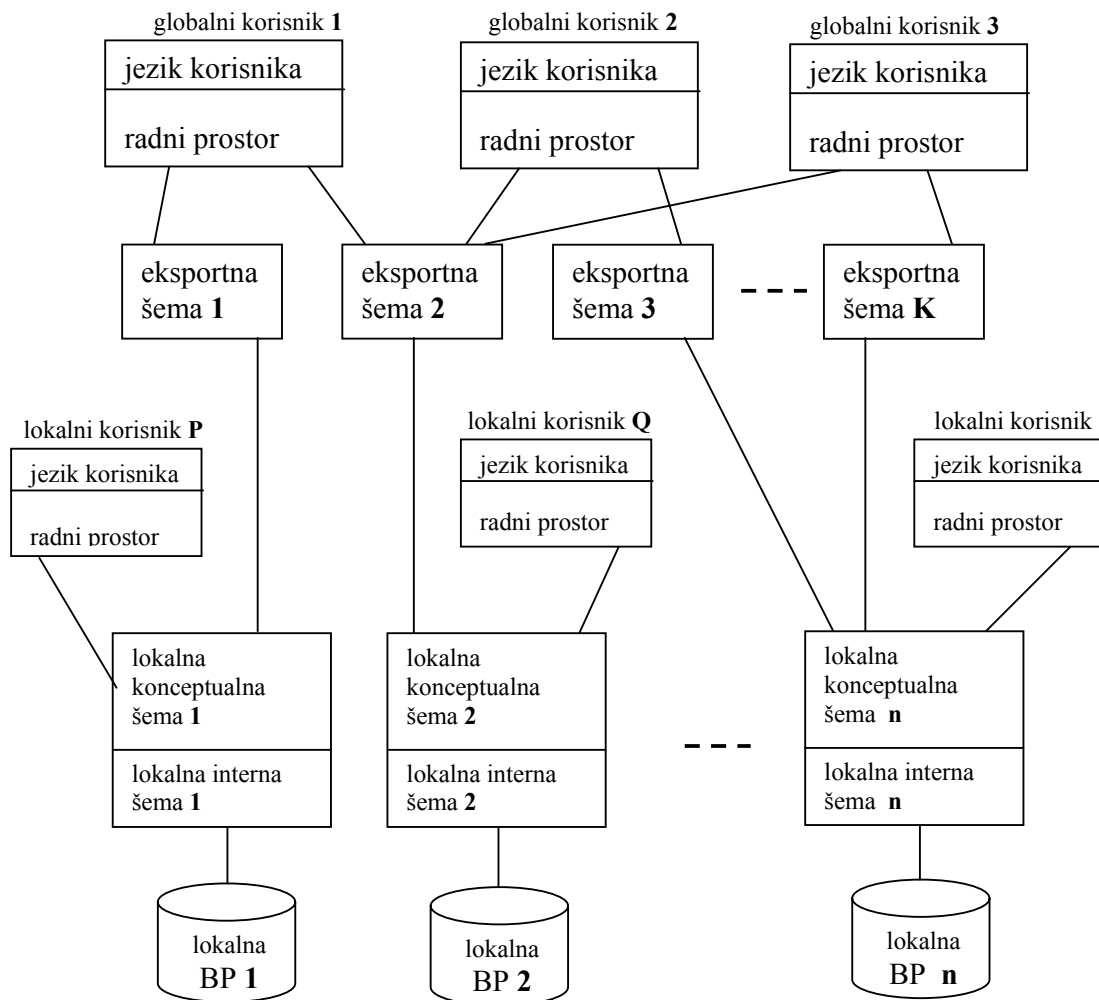
Federalizovani MDBMS

- Jedna od arhitektura DDBMS, MDBMS tipa, sastavljena od komponentnih DBMS-ova.
- Hibrid između distribuiranog (za globalnog korisnika) i centralizovanog (za lokalnog korisnika) DBMS.

Federalizovane BP



- Federacija labavo vezanih autonomnih DB servera različitih isporučilaca koji komuniciraju po principu najmanjeg zajedničkog sadržaja.
- DBA lokalnog DBMS-a specificira eksportnu šemu i autorizuje pristup delovima lok. BP spoljnim (globalnim) korisnicima.

Arhitektura federalizovanog MDBMS- labavo vezana varijanta

Klijent-server sistemi

- Klasifikacija c/s arhitektura
- Tipovi c/s sistema
- Komponente distribuiranih IS
- Klijent-server arhitektura predstavlja arhitekturu koja se sastoji od dve platforme (h/w,os) koje imaju uloge klijenata i servera.
//platforme povezane kom. mrežom
- Klijent je proces koji inicira zahtev za uslugama (serverima). Ima aktivnu ulogu.
- Server je proces koji odgovara na zahtev, obezbeđuje servise za druge platforme. Ima pasivnu ulogu.

Klasifikacija c/s arhitektura

- prema hijerarhiji
- prema lokaciji
- prema nameni/funkciji

Prema hijerarhiji:

- Dvonivojska c/s arhitektura
- Višenivojska c/s arhitektura

Dvonivojska c/s arhitektura

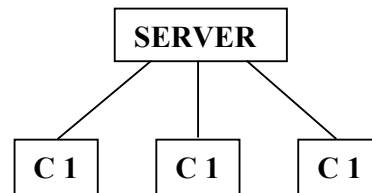
- klasična
- Server-centrična
- Klijent-centrična
- Mrežna (od sloja do sloja P2P)

C/S arhitektura sa aspekta hijerarhije-dvonivojska

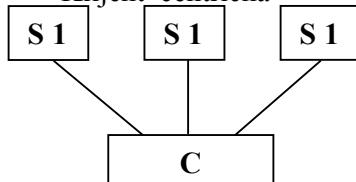
- klasična



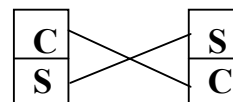
- Server-centrična



- Klijent-centrična

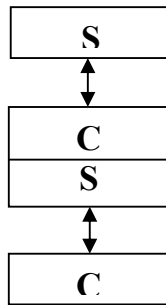
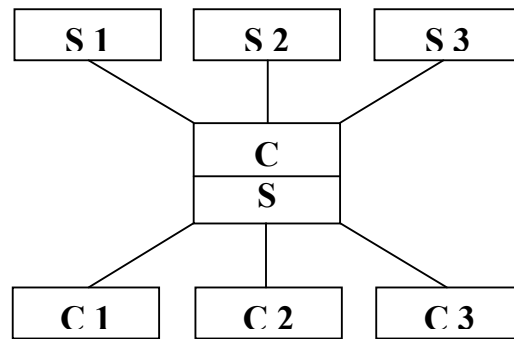


- Mrežna



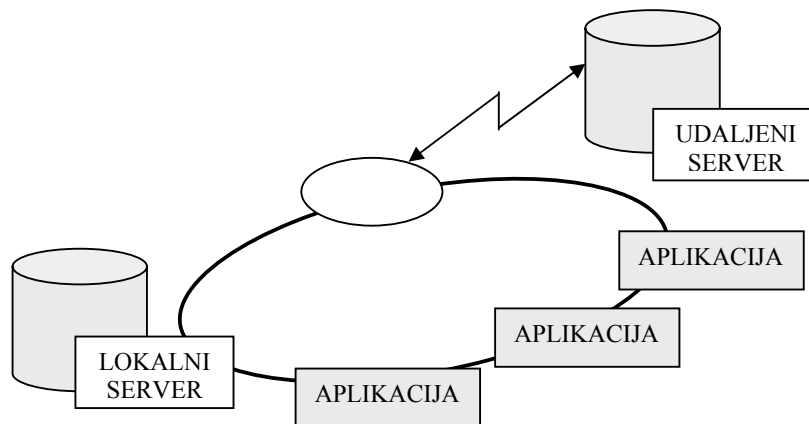
c/s arhitektura sa aspekta hijerarhije– visenivojska:

- “Gateway” tipa
- “Routing–Gateway” tipa

“Gateway” tipa**“Routing–Gateway” tipa****Prema lokaciji**

Sa aspekta lokacije, u okviru c/s arhitektura, može se govoriti o:

- Lokalnim serverima
- Udaljenim serverima
- Svi serveri na jednom LAN–u su **lokalni serveri**, njihovi tipični korisnici su Radne Grupe (workgroup), povezivanje zajedničkim poslom i podacima
- Lok. Server koji opslužuje radnu grupu se naziva server radne grupe (WG server).
- Server udaljen u odnosu na dati LAN, koji se pojavljuje kao server za njegove klijente, se naziva **udaljeni server**.
- Servewr koji deli podatke i servise preko organizacionih granica WG, tj. služi potrebama celog preduzeća, se naziva **Kompanijski (Enterprise) server**.

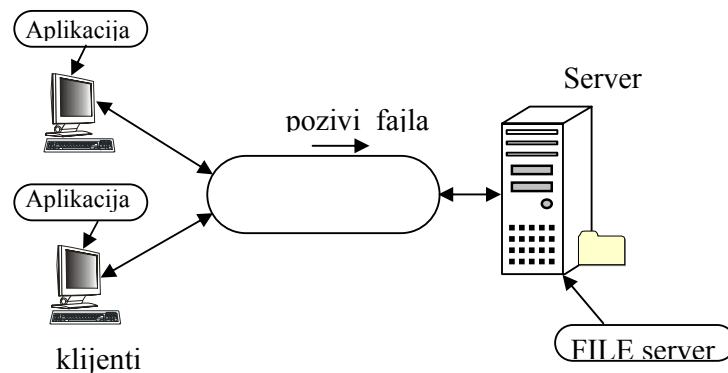


Klasifikacija c/s sistema prema nameni, funkciji servera

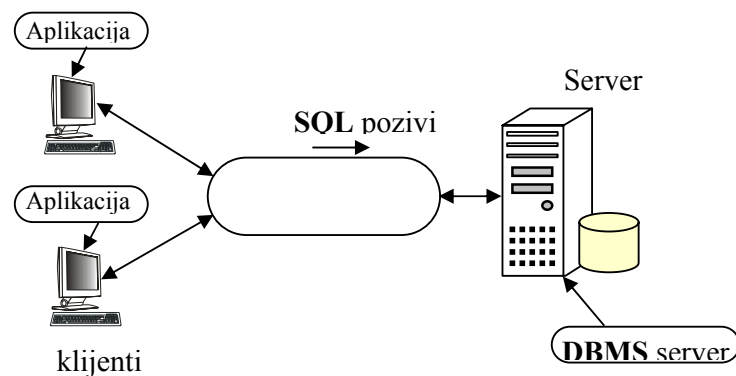
- File serveri
- DB serveri
- Transakcioni serveri
- Serverske grupe
- Objektni aplikacioni serveri
- Web aplikacioni serveri

File serveri

- Zajedničko korišćenje fajlova
- Klijent šalje zahtev za rekordima u fajlu.
- Intezivan saobraćaj na mreži jer FS pošalje ceo fajl koji klijent lokalno potražuje.
- Dobro za deljenje repozitorijuma dokumenata, slika, inž. crteža i drugih velikih data-objekata.

C/S sistem sa file serverom**DB serveri**

- Klijent šalje SQL zahteve kao poruke SQL serveru.
- Rezultati svake SQL naredbe se vraćaju preko mreže.
- Kod koji obrađuje SQL zahteve i podaci se nalaze na istoj mašini.
- Na strani klijentaa se mora napisati kod za klijnt. aplikacije, ili se mora kupiti upitni alat.
- Osnova za DS sisteme i DW.

C/S sistem sa DB serverom

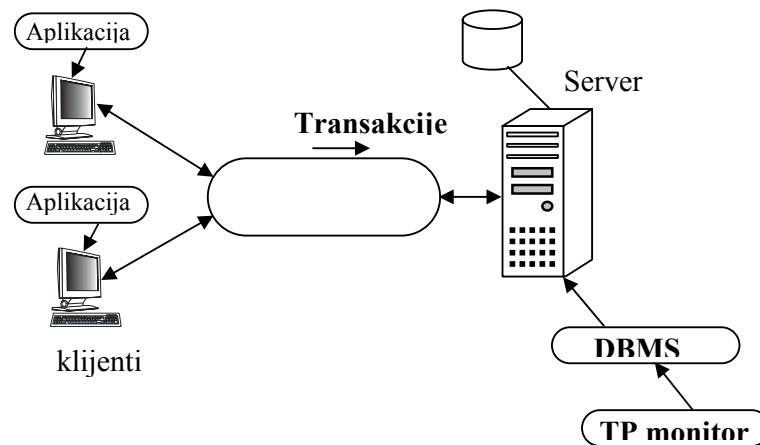
Transakcioni serveri

- Sa transakcionim serverom klijent pokreće udaljene procedure (servise) koji se nalaze na SQL (DB) serveru.
- Te udaljene proc. izvršavaju grupu SQL instrukcija.
- Saobraćaj po mreži se sastoji od jedne proste (zahtev/odgovor) poruke, tj SQL komande su agregirane u transakcije.
- Sa TS za c/s aplikacije treba napisati kod sa obe strane, i klijentske i serverske.
- Klijenti su obično GUI.
//GUI–Graphic User Interface
- Server sa aplikacijama OLTP tipa, tj. SQL transakcija na BP.

Dve forme OLTP-a:

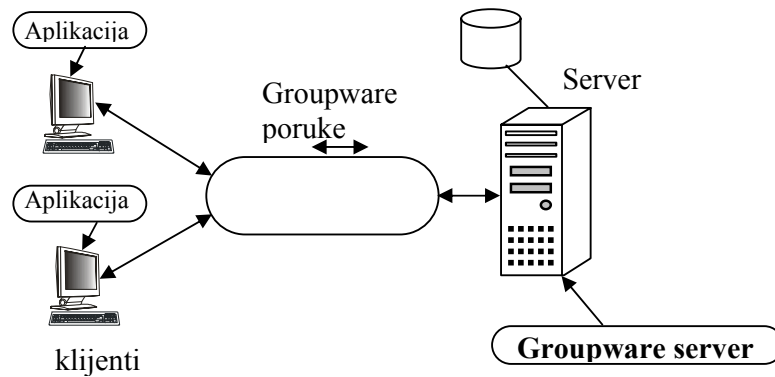
- **TP-lite**, na bazi **store procedura**.
- **TP-heavy**, na bazi **TP Monitora**.

C/S sistem sa transakcionim serverom



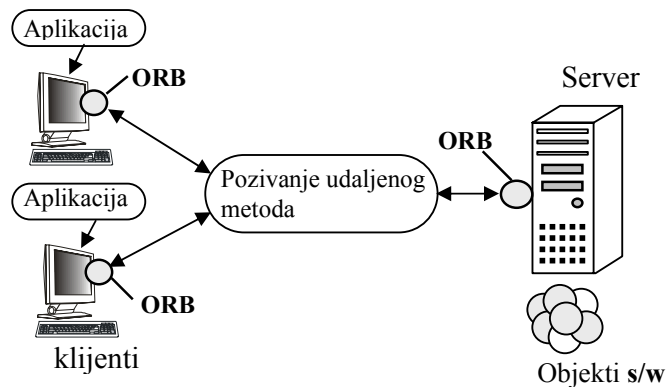
Serveri grupe (Groupware)

- GW adresira upravljanje polu struktuiranih informacija tipa: tekst, slika, mail, bulletin board, tok posla (workflow).
- C/S sistemi ovoga tipa stavljaju ljude u direktan kontakt sa drugim ljudima.
- U ovu kategoriju spadaju i :
 - Document mngm. Syst.
 - Imaging syst.
 - Myltparty apl.
 - Workflow
- Mnogi GW produkti danas koriste e-mail kao standardni srednji sloj.
- Često se danas Internet posmatra kao MW platforma za groupware.

C/S sistem sa Serverom grupe**Objektni aplikacioni serveri**

- Sa objektnim serverom, c/s aplikacija je napisana kao skup objekata koji komuniciraju.
- Klijentski objekti komuniciraju sa serverskim objektima korišćenjem ORB-a, tj. Klijent pokreće/poziva metod na udaljenom objektu.
- ORB locira instancu te kalse serverskog objekta, pokreće zahtevani metod, vraća rezultat klijentskom objektu.

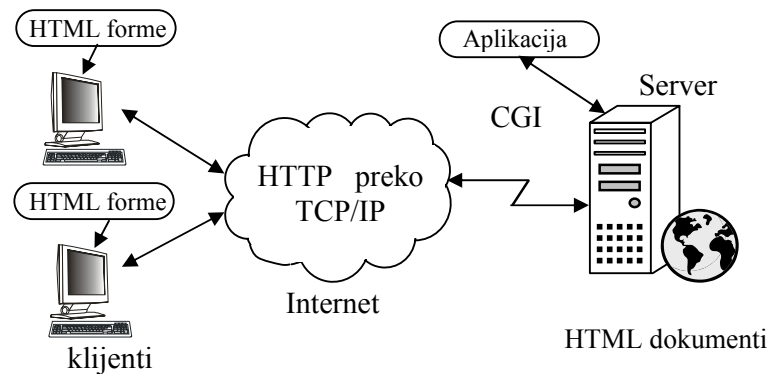
// ORB – Object Requested Broker – Objektni Raspodeljivač
RMI – Remote Method Invocation

C/S sitem sa Distribuiranim objektima

Web aplikacioni serveri

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

C/S sistem sa Web aplikacionim serverom



Komponente distribuiranih IS

- Podaci
 - Obrada
 - Prezentacija
 - Korisnički interfejs
 - Upravljanje prezentacijom
 - Niz zajedničkih servisa
- Do kog stepena vršiti distribiciju ovih komponenata DIS?
 Particioniranje aplikacije, sistema.

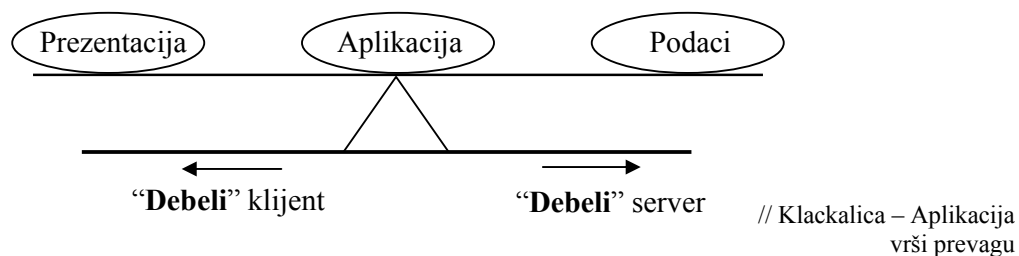
Arhitektura aplikacija

Dvoslojna arhitektura:

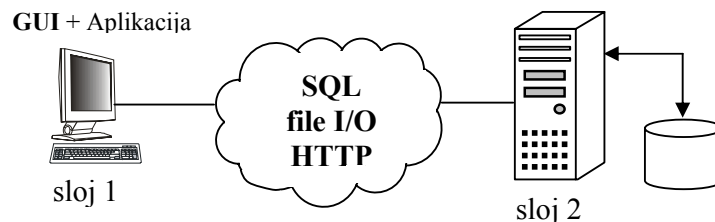
- Debeli klijent
 - Na klijentu se prezentacija i obrada
 - Na serveru su podaci
 Veza: RDA ili RFA
 - RDA–Remote Data Access, RFA–Remote File Access
- Debeli server
 - Na klijentu je prezentacija
 - Na serveru su podaci i obrada

Veza: poziv memorisane procedure

Dvoslojna arhitektura c/s sistema



Dvoslojna fizička arhitektura c/s sistema (h/w, s/w)



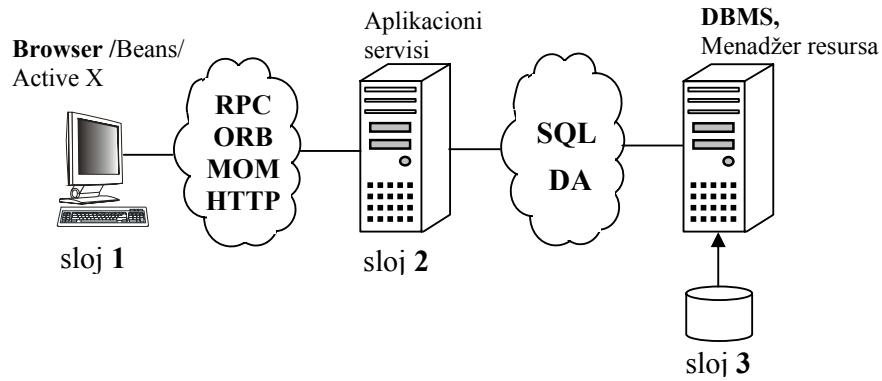
// Varijanta sa debelim klijentom
 Za debeli server, aplikacija bi bila
 na strani servera, a sve ostalo je isto

Arhitekture aplikacija

Troslojna arhitektura:

1. Sloj-Prezentaciona komponenta
veza: RDA, RPC, poruke, obrada transakcija
2. Sloj-zajednički aplikacioni servisi (AS)
veza: RDA, obr. Transakc., RPC, poruke
3. Sloj-Zajednički servisi podataka
 - Veza 2. i 3. sloja je ili RDA ili RFA ili obr. transakcija.

Troslojna fizička arhitektura c/s sistema (h/w, s/w)



Klijent-server sistemi-2

Sadržaj

- ☐ Komponente distribuiranih sistema
- ☐ Dvoslojna vs Troslojna c/s arhitektura
- ☐ Alokacija funkcija na elemente c/s sistema i funkcionalna dekompozicija.
- ☐ Tehnološki aspekti c/s sistema

Komponente distribuiranih IS

- ☐ Podaci
- ☐ Obrada
- ☐ Prezentacija
 - Korisnički interfejs
 - Upravljanje prezentacijom
- ☐ Niz zajedničkih servisa

Do kog stepena vršiti distribuciju ovih komponenata DIS ?

Particioniranje aplikacije (sistema).

Komponente DIS: podaci

Ova **komponenta** se bavi strukturama i funkcijama za čuvanje i manipulisanje podacima.

Sastoji se od brojnih objekata podataka (data objects) koji reprezentuju razne tipove medija: RDB, grafički fajl, audio fajl, ili multimedia tok podataka (data stream)

Komponente DIS: obrada

Ova **komponenta** se bavi obradom objekata podataka.

Može sadržati **procesne objekte** kao što su: Viewers, browser, search engines, aplikativno-specifična logika.

Komponente DIS: prezentacija

Ova komponenta se bavi vizualizacijom podataka korisnika i prihvatom njegove interakcije, na dva nivoa:

- **Korisnički interfejs**, aspekti koje korisnik direktno vidi. Sastoji se od standardnog skupa prikaza, konsistentnost.
- **Upravljanje prezentacijom**, UI manager obezbeđuje osnovne operacije koje se koriste za pravljenje i upravljanje UI od strane aplikacije (korisnika).

UIM S/W za upravljanje prezentacijom

obezbeđuje tri osnovne funkcije:

- **Servise prikaza**, koji upravljaju svim UI uređajima (ekran, miš, tastatura, video kamera, mikrofona, zvučnici).
- **Upravljanje dijalogom**, obezbeđuje mehanizme za kreiranje i manipulaciju sa UI, kao i event-handling funkcije.
- **API**, obezbeđuje svakoj aplikaciji interfejse putem kojih se prave aplikaciono specifični interfejsi.

Primer: UIM s/w je MS Windows s/w

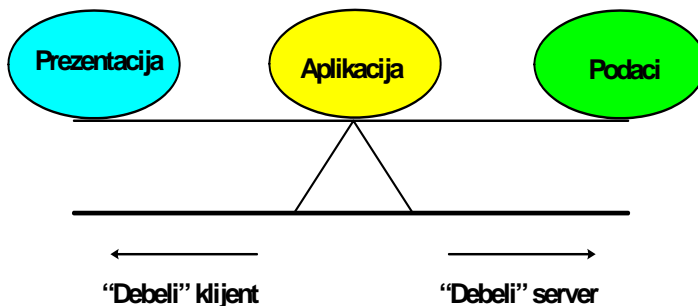
Komponente DIS: zajednički servisi

- Povezivanja
- Identifikacije (Naming i adressing) komponenata
- Administracije i upravljanja.
- Sigurnosti

Obično su deo distribuirane IT infrastrukture.

Dvoslojna arhitektura aplikacija

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

Dvoslojna arhitektura c/s sistema**Tronivojska, 3-slojna(3 tier)arh**

Termin prilično zloupotrebljen!

Ranije: PC, Department server, Enterprise server

Potom: Klijent, Lokalna BP, Enterprise BP

Danas: Klijent-Apl. server-DB server

Troslojna arhitekture aplikacija

- 1.Sloj-**Prezentaciona komponenta.**
veza: RDA, RPC, poruke, obrada transakcija.
- 2.Sloj-**Zajednički aplikacioni servisi (AS).**
veza: RDA, obr. transakc., RPC, poruke
- 3.Sloj-**Zajednički servisi podataka.**

Komparacija arhitektura-1

Svojstvo	2-slojna	3-slojna
Administracija sistema	Kompleksna	Manje kompleksna
Sigurnost	Niska	Visoka
Performanse	Slabe	Dobre
Skalabilnost	Slaba	Odlična
Lakoća razvoja	Visoka	Postaje bolja

- 2-slojna arh. je **kompleksnija**, kao posledica više "logike" na klijentu kojom treba upravljati.
- 3-slojna je manje kompleksna jer se može centralno upravljati na AS.
- **Sigurnost**(2slojne) je na nivou podataka, dok se kod 3slojne može finije podešavati a nivou servisa, metoda ili tipa objekta).
- 2-slojna arh. ima slabije **perf.** kao posledica intenzivnijeg SQL saobraćaja. Lošija **skalabilnost** kao posledica ograničenih mogućnosti upravljanja kom. linkovima klijenta.
- 3-slojna, koncentriše dolazeće sesije, može distribuirati opterećenje na višestruke servere.

Komparacija arhitektura-2

Svojstvo	2-slojna	3-slojna
Enkapsulacija podataka	Niska	Visoka
Ponovno korišćenje aplikacije	Slabo	Odlično
Server-to -server infrastruktura	Ne	Da
Podrška za Internet	Slaba	Odlična
Izbor mogućnosti komunikacija	Nema	Ima

Dvoslojna arhitektura

Osnovna svojstva:

Jednostavnost, brzina realizacije, posebno korišćenjem vizuelnih alata.

⇒ **podesno za Apl. na nivou radne grupe/oddeljenja** (Department):

DSS, manje groupware, jednostavni web publishing

Troslojna arhitektura

Kada se prevaziđu granice odeljenskog LAN-a:

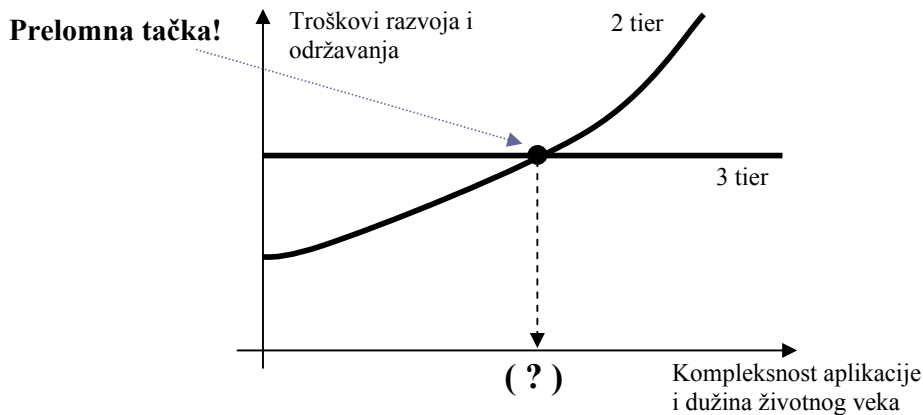
- $N > 300$ konkurentnih korisnika
- Kompleksne apl. sa > 50 apl. klasa, servisa
- ≥ 2 DBMS, heterogena izvora podataka
- Veliko transakciono opterećenje ($> 50\,000$ transakcija/dan)
- Aplikacije na različitim programskim jezicima

1998. 33% svih c/s apl. su koristile 3-slojni model

Dvoslojna vs troslojna arh.

Dijagram:

- Troškovi razvoja i održavanja IS u odnosu na
- Kompleksnost aplikacije i dužinu životnog veka apl.



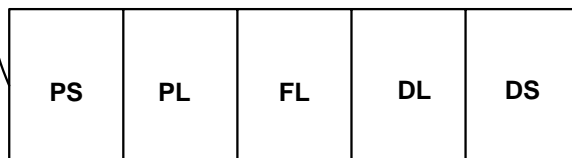
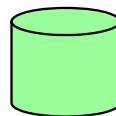
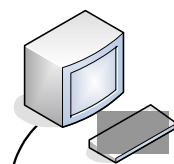
Alokacija funkcija na elemente

- Moguće različite klasifikacije, pristupi.
- Brojna (zbunjujuća?) terminologija.
- Prikazaćemo dve, stariju i noviju, moguće klasifikacije.

Osnovna struktura aplikacije (IS)

Prostor prezentacije

Prostor podataka



- PS-prezentacioni servisi
- PL-prezentaciona logika
- FL-funkcionalna logika
- DL-logika podataka
- DS-sevisi podataka

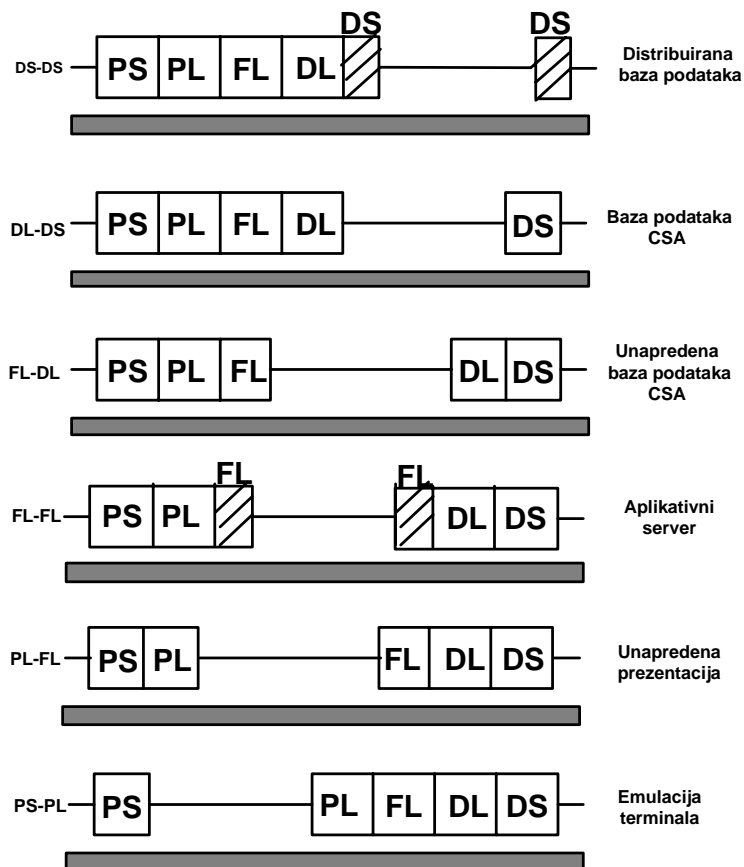
Upravljanje transakcijama

- ☐ **PS**-omogućuju prikaz inf. korisniku, određuje izgled (look) apl. Ranije: protokoli tipa VT 100, 3270, danas x-window.
- ☐ **PL**-dopunjuje PS, određuje ponašanje (feel) apl. Ranije tipa **DA/NE**, danas tipično **skup procedura** koje se pozivaju kao reakcija na određeni događaj (event).
- ☐ **FL**-implementira **poslovna pravila** aplikacije.
- ☐ **DL**-određuje ponašanje obj. pod. u okviru resursa pod. Ranije, sakriven u kodu (formati rekorda pod.), danas, relacioni pogled ili store procedura.
- ☐ **DS**-upravlja definicijom strukture i manipulacijom vrednostima za obj. pod. u resursima podataka. Ranije se realizovali od strane file-managera(OS), danas DS omogućava SQL interfejs prema RDBMS kernelu.

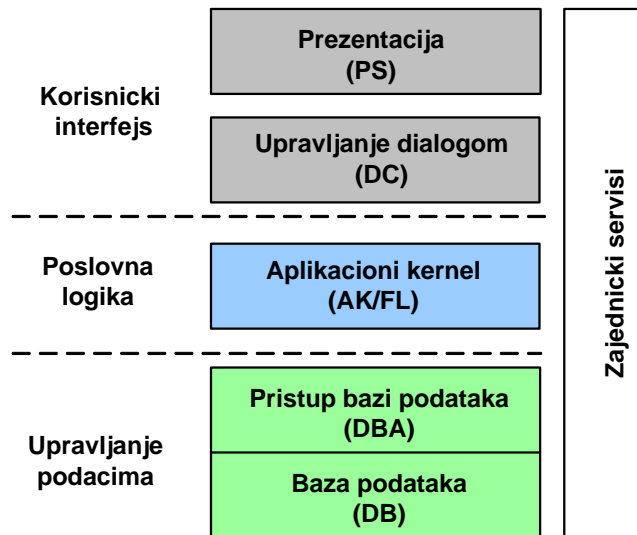
Podela na nivou: Naziv:

- ☐ **DS-DS**: distribuirana BP
- ☐ **DL-DS**: BP klijent server arhitekture
- ☐ **FL-DL**: unapređena BP C/S Arh.
- ☐ **FL-FL**: aplikativni server
- ☐ **PL-FL**: unapređena prezentacija
- ☐ **PS-PL**: emulacija terminala

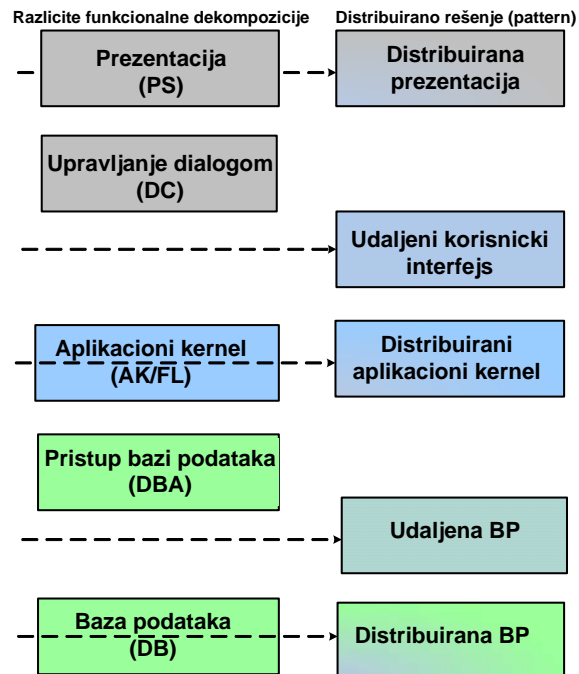
Moguća alokacija funkcija



Alternativni načini particioniranja c/s aplikacije



Moguća distribuirana rešenja



Distribucija nudi šire mogućnosti za dobar dizajn sistema, ali istovremeno i usložnjava razvoj odgovarajuće arhitekture jer uvodi nove aspekte, kriterijume u razmatranje, npr.:

Poslovne potrebe, stil obrade, performanse, sigurnost, konsistentnost, troškovi distribucije, reusability,...

Distribuirana prezentacija

- Rešenje razdvaja sistem u okviru prezentacione komponente, čiji se delovi odvojeno procesiraju.
- Rešenje omogućava jednostavnu implementaciju i veoma tanak klijent.

Primeri: host-terminali(IBM 3270)

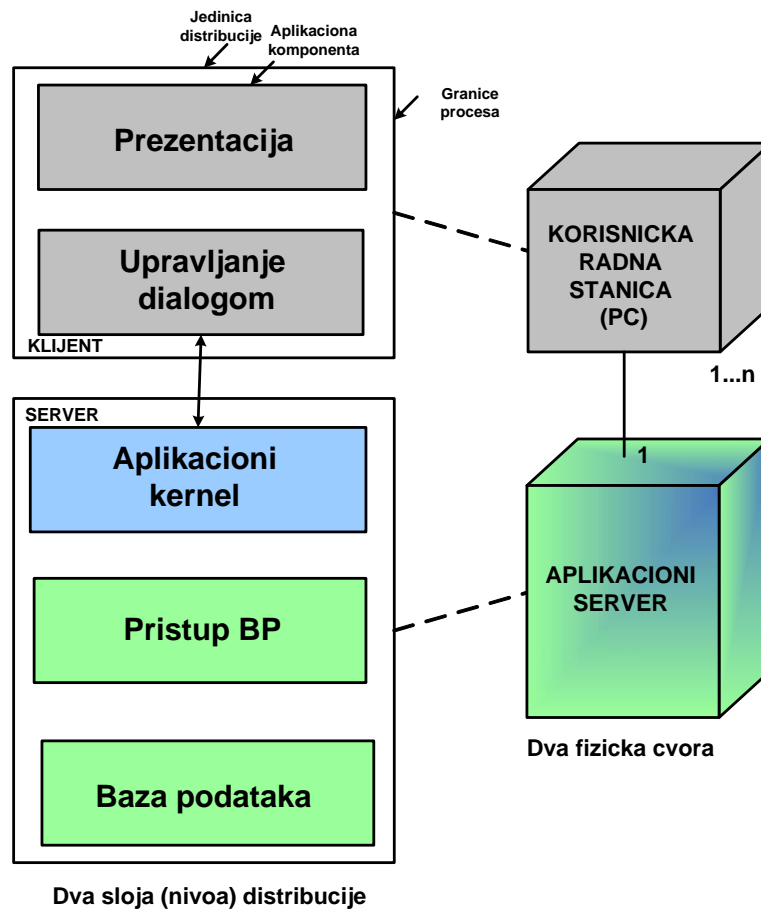
mrežni računar(NC)

Internet (Browser)

Udaljeni korisnički interfejs (RUI)

Celokupni korisnički interfejs postaje jedinica distribucije i ponaša se kao klijent aplikacionog kernela na strani servera.

Aplikaciono zavisni pristupni protokol (MW): RPC, CORBA, RMI (Java).

2-slojna c/s arh. sa udaljenim korisničkim interfejsom (RUI)**Distribuirani aplikacioni kernel**

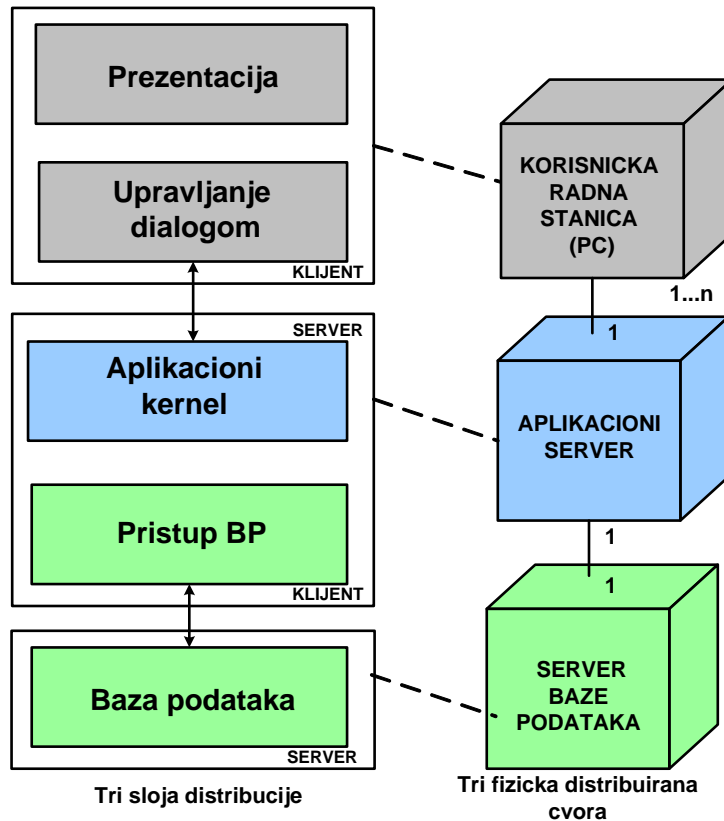
Ovo rešenje razdvaja apl. kernel u dva dela koja se odvojeno obrađuju. Rešenje je izazovno za slučaj da transakcija prevazilazi granice proces (distribuirana obrada transakcija).

Udaljena baza podataka (RDB)

-Baza podataka predstavlja komponentu IS sa specijalnim zahtevima u pogledu okruženja.

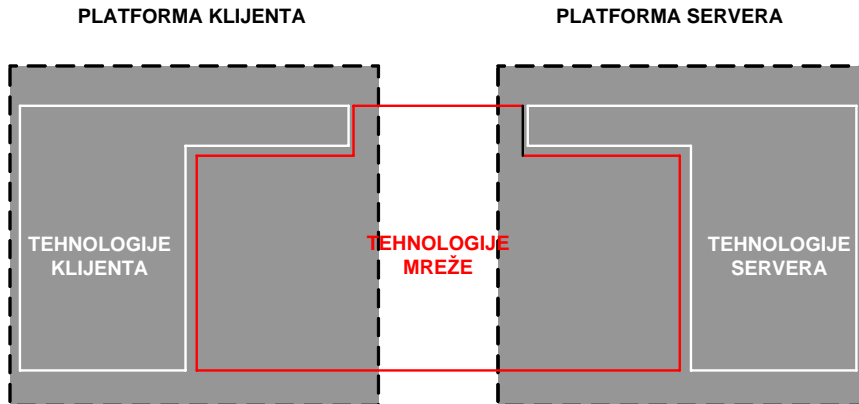
-Ukoliko nekoliko aplikacija radi nad istom BP podesno je BP komponentu locirati na poseban čvor mreže.

-Veza debelog klijenta (2slojna) sa BP putem RDA protokola.

3-slojna c/s arh. sa udaljenom BP**Distribuirana baza podataka**

Baza podataka se dekomponuje u odvojene DB komponente, koje interaguju putem sredstava interprocesorske komunikacije (IPC).

Osnovne c/s tehnologije

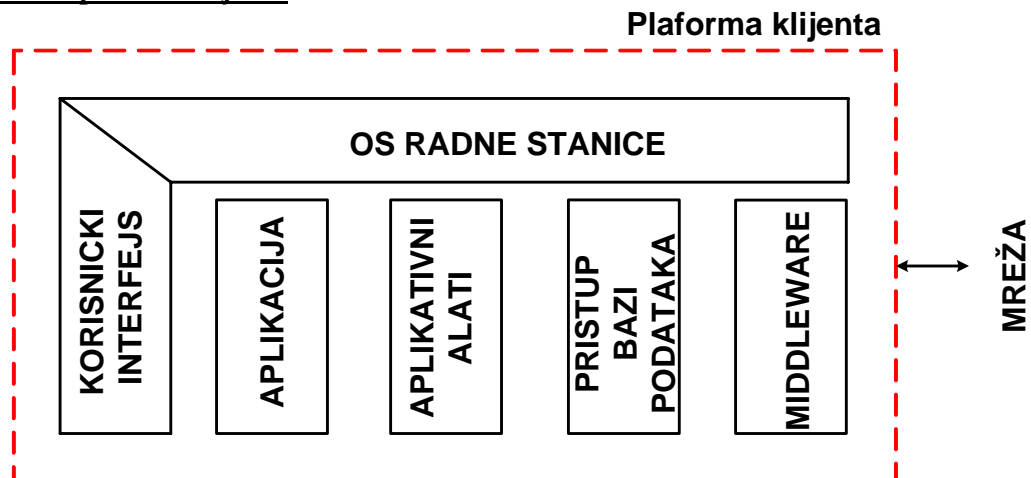


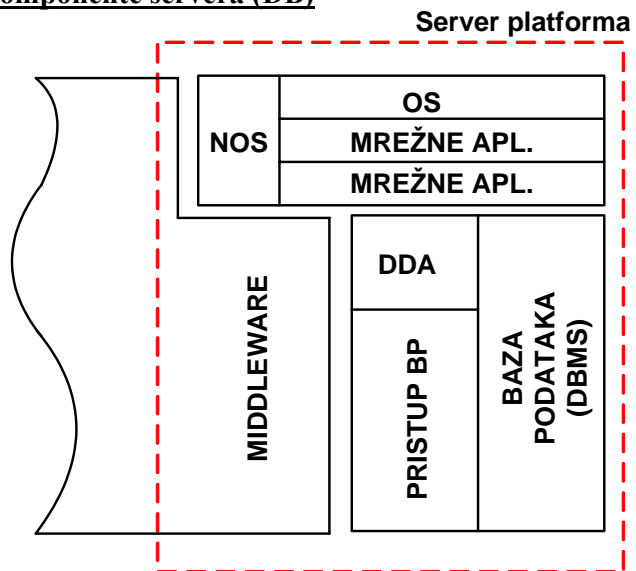
- Granice platforme se ne poklapaju sa granicama tehnologije

Tehnološki elementi c/s sistema



s/w komponente klijenta



s/w komponente servera (DB)**Tehnološki elementi c/s sistema**

- Prikazani tehnološki elementi su definisani početkom 90 tih godina.
- Važe i danas ali za DB-bazirane c/s sisteme.

Novija, alternativna, klasifikacija se prikazuje u nastavku.

Tehnološki aspekti c/s sistema

Sadržaj:

- ☐ Tehnologije klijenta
- ☐ Tehnologije servera
- ☐ Alati za razvoj aplikacija

Tehnološki aspekti c/s sistema

- Tehnologije klijenta
 - Tehnologije servera
 - Tehnologije mreža
- Generalizacija arhitekture sa početka '90ih

Tehnološki aspekti-generalizacija

- ☐ Kao posledica ukupnog napretka IT, novih potreba (npr. DSM komponenta)
- ☐ Kao posledica evolucije c/s sistema, od DB-servera do servera različitih servisa.
- ☐ Kao posledica razvoja pojma i sadržaja **middleware-a**, tj. srednjeg sloja.

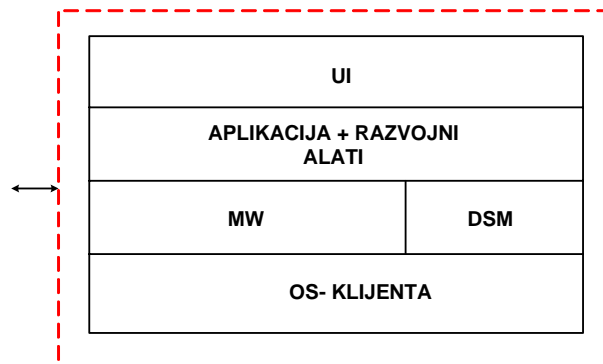
s/w arhitektura c/s sistema-generalizacija

DSM	DSM	DSM
UI (GUI, OOUI, Web Browser)	Servisno-specificni MW	SERVISI: - DBMS - OLTP - GW - Web - Objekti
Aplikacija		
OS	NOS-sevisi <input type="checkbox"/> Direktorijum <input type="checkbox"/> Distribucija vremena <input type="checkbox"/> Sigurnosti	OS
	Komunikacioni protokoli (transport)	

Tehnologije klijenta

- ❑ 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

s/w arhitektura klijenta



Hardverska platforma klijenta

Predstavlja osnovnu komponentu sistema na koju se oslanjaju sve ostale. Sadrži:

- Centralni procesor (CPU)
 - Memoriju sa direktnim pristupom (RAM)
 - Masovnu memoriju (diskovi, CD, tape,...)
 - Ulazne/izlazne uređaje(tastatura, miš,...)
 - Monitor
-
- CPU: različite arhitekture (CICS, RISC)
 - CPU: 32, 64 bita
 - U širokom rasponu od embeded uređaja do super klijenata.

Operativni sistem klijenta

Osnovna namena OS klijenta je da aplikacijama omogući pristup h/w resursima i upravljanje interfejsima između ws 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 značajno 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 omogući interakciju sa samom aplikacijom i OS na intuitivan i jednostavan grafički način, konzistentan za sve aplikacije iz toga okruženja.

Operativni sistem klijenta: GUI (graphical user interface) aspekti

- GUI može biti ugnježđen u OS, tj. tretiran kao deo OS, npr. Mac OS, Windows NT, ili
- obezbeđen kao proširenje OS, npr. MS Window/MS DOS, ili različita

UNIX GUI sredstva (OSF/Motif, Sun Open Look,...).

X-window nije UNIX GUI nego API niskog nivoa i prateći protokol

Hardverska nezavisnost OS

- Portabilnost OS je veoma značajna.
- UNIX se smatra danas najotvorenijim i najportabilnijim.
- Ranije, brojne različite mplementacije UNIX-a, danas, standardizovan.

Važan novi zahtev: mogućnost izvršavanja mobilnog koda, tj. postojanje Java VM na Klijentu.

OS klijenta: primeri

- ☐ Windows 95/98, 16 bitni, nad DOS-om!
- ☐ Windows 2000 (Win NT 5.0)/XP, 32 bitni
- ☐ Windows NT 4.0 workstation, NT klijenta, 32 bitni
- ☐ Apple Mac OS 8 ⇒ OS X
- ☐ IBM OS/2 Warp
- ☐ Linux (Red Hat, Caldera, ...)
- ☐ Windows CE za "embedded" uređaje
- ☐ Embedded Java OS (+Jini, mob. upravlj. agent)

- ☐ Windows NT klijenta (32 bitni) i na njemu bazirani os:
 - Na početku se smatrao zahtevnim u pogledu resursa
 - Manji broj raspoloživih drajvera
 - Veća cena (250 USD vs. 100 USD Win 98)
- ☐ Apple Mac OS X, savremeni klijent OS, podrška za Java VM, ali potrebni novi drajveri za veći deo h/w

- ☐ Linux, iz Open source priče
 - najpouzdanija verzija UNIX-a za Intel
 - jeftin
 - do skora radio samo na serverima
 - za "pogon" Apache Web servera
 - nepodržava MS Office, alternativa:
 - Star Office
 - Open Office

Srednji sloj (MW) i DSM

- ☐ U ovom segmentu su se tehnologija, sadržaj i terminologija bitno menjali.
- ☐ Middleware je ključni deo C/S infrastrukture.
- ☐ MW takođe poseduje svoju DSM komponentu (s/w agent)

MW se izvršava i na klijentu i na serveru, deli se u tri kategorije:

- transportni stack
 - mrežni OS (NOS)
 - servisno-specifični MW (MW višeg nivoa)
- } (MW nižeg nivoa)

DSM – Distributed system management element

- Ova aplikacija radi u svakom čvoru mreže sa ciljem nadzora i upravljanja.
- Upravljačka WS prikuplja informacije od svih svojih agenata i grafički ih prikazuje.
- WS može dati nalog agentima da u njeno ime obave određenu akciju.
- Praktično autonomna mreža u okviru mreže!
- Tipično se koristi **SNMP** upravljački protokol (Simple Network Management Protocol)

UI klijenta

Tipovi UI klijenta:

- **Ne GUI klijent**
- **GUI klijent**
- **OO UI klijent**
- **Compound dokumenti**
- **3D compound dokumenti**

Ne GUI klijent

Različiti uređaji koji ne traže, ili traže minimalnu ljudsku interakciju. Npr.:

- Fax klijent
- Barkod čitač
- ATM //ATM–Automatic Tele Machine–Bankomat
- Celularni telefon
- roboti / manipulatori
-

GUI klijent

- **GUI aplikacije** se sastoji od jedne **ikone** i primarnog prozora sa meni barom.
- Pomoćni zadaci-dopunski prozori. Korisnik mora pratiti krutu strukturu zadataka.
- **Ikona** predstavlja aplikaciju u radu.
- **Meni** obezbeđuje osnovni metod kretanja kroz aplikaciju.
- Prvo se bira objekat a potom akcija iz meni bara.

GUI klijent-primeri

- ☐ Windows 3.X
 - ☐ Proste Web stranice
 - ☐ OSF Motif X-windows
- Pojavili se 1989.

OO UI klijent

- **OOUI aplikacije** se sastoji od **skupa** korisničkih **objekata** koji kooperiraju.
- **Ikone** predstavljaju objekte kojima se može direktno manipulirati.
- **Prozor** je pogled u unutrašnjost objekta. Svaki objekat ima kontekstni meni.
- **Kretanje kroz aplikaciju** ili između apl. se vrši direktnom manipulacijom objektima (drag-and-drop).

OO UI klijent-primeri

- ☐ Windows 98
- ☐ Apple Mac OS
- ☐ Web pages+**JB** (Java Beans)

Compound dokumenti

- Compound document framework su najnovija OOUI tehnologija (OOUI na stereoidima)
- Svaki vizualni objekat na ekranu je ''živa''komponenta, kojoj se može menjati veličina, položaj, videti sadržaj, nasleđivati svojstva,...
- Dokument postaje virtualni svet naseljen komponentama.

Mesta za isporuku (shippable place)

- **Mesto** je vizualna grupa povezanih komponentata.
- **Shippable place** je je mobilni kontejner (tj. mogu ugnjezditi druge komponente) za komponente.
- Mesta omogućavaju serverima da automatski ažuriraju desktopove svojih klijenata,tj. **vizualno mesto** na klijentu na kome se može prikazati info. u realnom vremenu (npr. baner, video feed)

Compound dokumenti-primeri

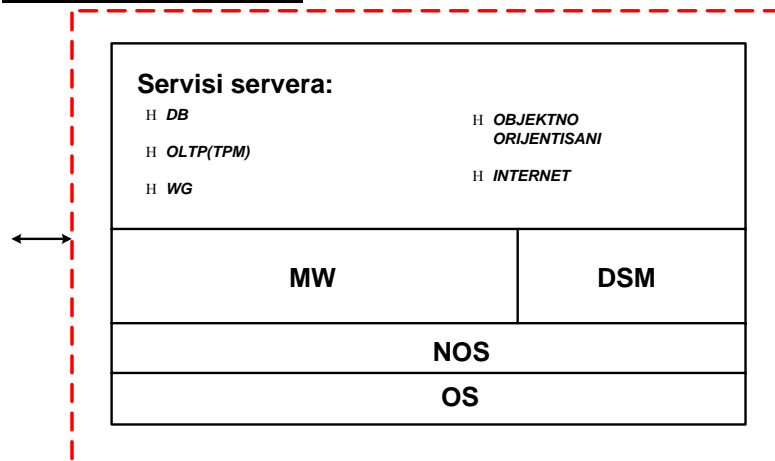
- Active X
- Java Beans
- Dynamic HTML

Tehnologije servera

- ☐ Kod c/s sistema, funkcije zajedničke za više klijenata se obavljaju na serveru.
- ☐ Važan **deljeni resurs** koji treba da obezbedi servise za veliki broj simultanih korisnika.
- ☐ Kod konfigurisanja treba voditi računa o pouzdanosti, performansama, fleksibilnosti,....

Mora se raspolagati sledećim komponentama:

- H/W platforma servera
- Operativni sistem servera
- Middleware (srednji sloj višeg i nižeg nivoa)
- DSM-upravljanje distribuiranim sistemom
- Specifičnim serverskim servisima

s/w arhitektura servera

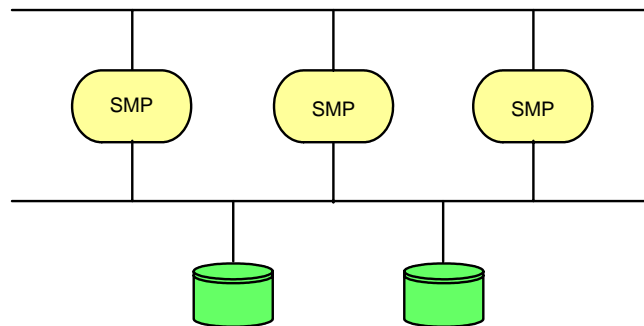
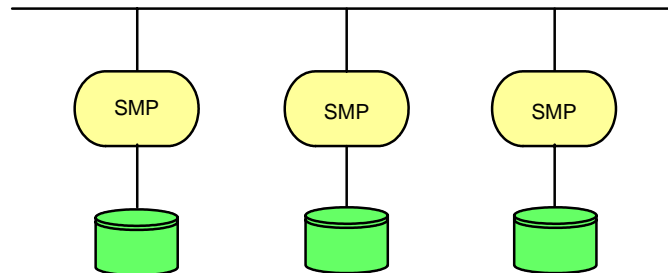
H/W platforma servera

- **PC-server**
- **AMP**-Asimetrični multi procesorski
- **SMP**-simetrični multi procesorski
- Multiserverski klasteri

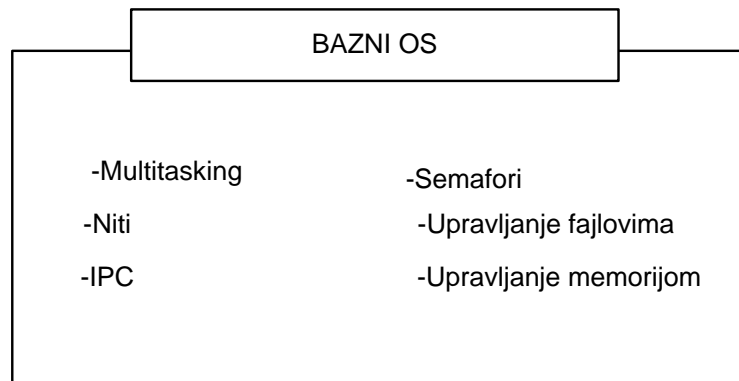
Multiserverski klasteri

U dve varijante:

- Sa deljenim diskovima (**Shared disk**)
- Bez deljenih diskova (**Shared nothing**)

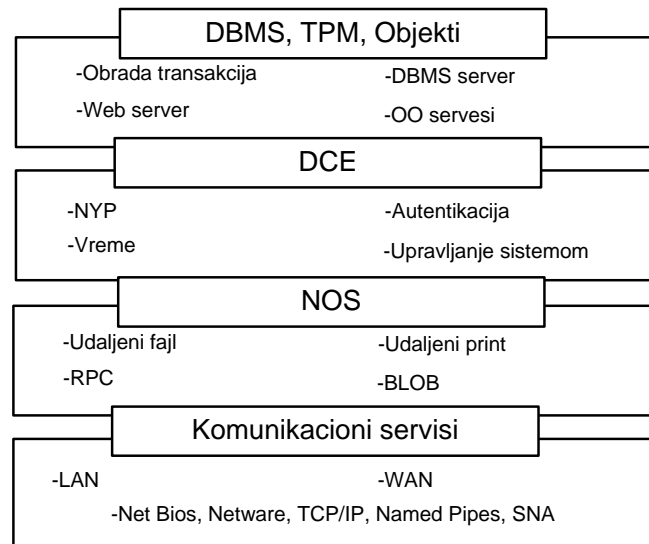
Sa deljenim diskovima (Shared disk)**Bez deljenih diskovima (Share nothing)****Operativni sistem servera**

- Aplikacijama (klijentima) obezbeđuje pristup resursima i upravlja periferijama.
- U okruženju distribuirane obrade f-je OS su ili bazne ili proširene. Bazni servisi su deo std. OS, dok su proširene obično dopunske modularne komponente.

Bazni servisi OS

// IPC—Inter Processing Communication

- Preventivni multitasking
- Prioritet taskova
- Semafori
- Međuprocesorska komunikacija IPC
- Loklna/udaljena IPC
- Niti (threads)
- Višekorisnički fajl sistem
- Efikasno upravljanje memorijom

Prošireni servisi OS

- Različiti komunikacioni protokoli
- NOS ekstenzije
- Binarni veliki objekti (BLOB)
- Globalni direktorijumi i žute stranice (YP)
- Objektno orijentisani servisi
- Upravljanje sistemom

OS servera-primeri

Za male i srednje sisteme:

- Novell: Net Ware ver.4, ver. 5
- NT server ver.4, ver.5
- UNIX (na Intelu: SCO,Solaris, Linux)

Za velike sisteme:

- UNIX (HP-UX, podržava klastere)
- IBM MVS (MF)

UNIX:

- Digital: OSF/1
- HP: HP-UX
- IBM: AIX
- Sun: Solaris
- Linux

Alati za razvoj C/S sistema

Sadržaj

- ☐ Tipovi alata za razvoj c/s aplikacija
- ☐ Izbor alata za razvoj
- ☐ Alati za razvoj-primeri

Alati za razvoj C/S sistema

- U situaciji kada postoji > 200 različitih alata za razvoj, izbor nije jednostavan!
- **Alat je bilo kakvo razvojno okruženje, kompajler, alat za izveštavanje, ili okvir za izgradnju ili “deployment” c/s aplikacija.**
- Većina kreira aplikaciju na strani klijenta.

Tipovi alata za razvoj

Pre svega iz ugla c/s sistema DB tipa

Jedna moguća podela:

1. 3GL alati
2. Specijalizovani alati
3. Više-platformski alati
4. “Smalltalk alati”
5. Fajl orijentisani i desktop DB alati

6. Alati za izveštavanje i OLAP
7. Generatori koda
8. CASE alati
9. Alati za particioniranje aplikacije
10. Web razvojni alati

Realni alati mogu spadati u nekoliko navedenih kategorija.

3 GL alati se mogu definisati kao:

- Tradicionalni opštenamenski programski jezici tipa: C/C++, Pascal, Cobol, Fortran.
- **Primeri:** MS Visual C++, Borland C++, Symantec C++, Borland Object Pascal...
- Nisu specifično pravljene za DB aplikacije
- Koriste kompajlere za pravljenje efikasnog izvršnog koda⇒**prednost u performansama**

Programski jezici

- Danas postoji najmanje **500 programskih jezika i dijalekata** koji su komercijalno ili javno raspoloživi, kao i još 200 razvijenih od strane firme za svoje sopstvene potrebe.
- Takozvani 500 LP(Language Problem), posebno pre Y2K. // Y2K – problem 2000. godine
- Procena instalisanog s/w u svetu (1998):
 - **Cobol: 30% (225 G LOC)** // LOC–Line of code
 - **C/C++: 20% (180 G LOC)**
 - **Assembler: 10% (140-220 G LOC)**
 - **Ostali jezici: 40% (280 GLOC)**

Specijalizovani alati, razvojna okruženja

- **IDE**-Integrisana razvojna okruženja.
- Napravljeni za specifičnu svrhu: kreiranja c/s DB aplikacija.
- Sadrže sve potrebno za projektovanje, izgradnju i instalaciju (deployment) aplikacije, tj. **poseduju: 4GL ili prog. jezik za vizualno programiranje, IDE, debugger, biblioteke objekata koje apl. programer može koristiti u apl.**

Specijalizovani alati, obično u obliku razvojnih okruženja

- IDE: screen painter, object browser, integrated debugger, code editor, ODBC,JDBC prema različitim BP, obično uvezani sa local DB serverom.
- Preferabilni za razvoj c/s aplikacija ⇔ brzina razvoja aplikacije.
- **Primeri:** Delphi, Power Builder, Visual Basic, Oracle Developer,...
- Većina koristi interpretere a ne kompajlere, performanse slabije, ali to se menja (VB, Power Builder)

Više-platformski alati

- **Cross-platform tools** su slični IDE-ovima, ali su sposobni da distribuiraju izvršne verzije aplikacije na bilo koji broj platformi.
- Tamo gde postoji heterogena mešavina klijentskih platformi, tj. mogućnost podrške većem broju OS i UI-ja.
- **Primeri:** JAM 7 (JYACC Corp.), Unify, Uniface (Compuware Corp.).

”Smalltalk alati”

- Posebna kategorija, čisti obj.orientisani koncepti.
- Koristi i relacione BP kao Smalltalk objekte (putem specifičnog wrapera).
- **Primeri:** Object Studio (VMark Softw. inc.), Visual Age for Smalltalk (IBM)....

Fajl orijentisani i desktop DB alati

Već malo zastarela kategorija (X-base alati)

Primeri: MS Visual Fox Pro, Access, Borland Visual dBase,...

Poseduju unutar alata mehanizme za povezivanje sa BP, za migraciju aplikacije (npr. Fox Pro), podataka iz file server sistema u c/s sistem.

Alati za izveštavanje i OLAP

- Tipično **SQL Report writeri** omoučavaju vizuelni dizajn izveštaja i generišu potrebni SQL kod.
- Alati za kreiranje specijalizovanih analitičkih aplikacija (**OLAP**) nad DW.
- Primeri SQL: Borland Report Smith, Crystal Report (Seagate Software),
OLAP: Power Play (Cognos)

Generatori koda

- U osnovi generišu **3 GL kod. Koji potom ide na kompajliranje.**
- **Primer:** Proto Gen (Protosoft)

CASE alati

- Danas veoma široka, raznorodna kategorija alata.
- Omoučavaju programerima da dizajniraju i instaliraju, kako aplikacije, tako i BP.
- Poseduju podsisteme za crtanje dijagrama, razumevanje sistema na logičkom nivou, pre generisanja apl. objekata i DB šema.

CASE alati Obično u dve kategorije:

- **CASE alati višeg nivoa**(modliranje)
 - **CASE alati nižeg nivoa**(generisanje koda, npr. skriptova za kreiranje tabela RDB)
 - ❑ **Primeri alata za modeliranje:**
 - Rational Rose (Rational Software), UML
 - ERwin (Logic Works)
- ⇒ generišu **apl. kod za VB, Power Builder**

Alati za partitioniranje aplikacije

- Ovi alati omogućavaju **dinamičko partitioniranje sistema** (aplikacije).
 - Prvo se apl. logički kreira a potom se partitionira na run-time obj. koji se instaliraju na proizvoljan broj raspoloživih servera.
 - Korišćenje **N-to slojnog modela**, tj. veći broj AS.
 - ❑ **Primeri:**
Forte (Forte Sofrtware⇒Sun), Dynasty (Dynasty Technologies Inc.), Cactus (Information Builders Inc.)
- Ali alati su proizvođački, cena!**

Alati za partitioniranje aplikacije vs. Načini partitioniranja aplikacija

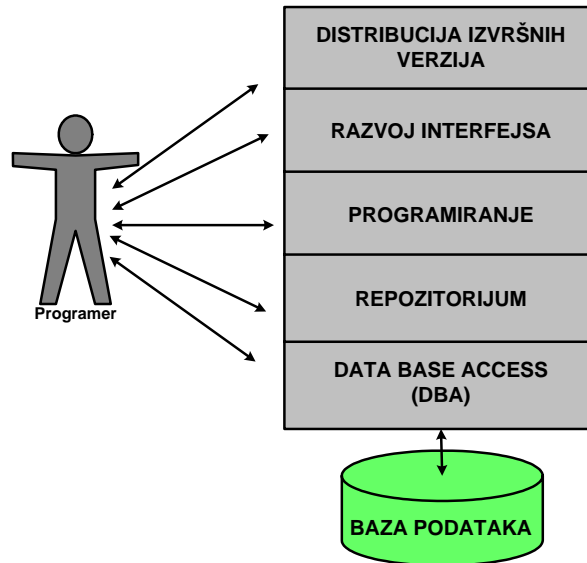
- **Statičko partitioniranje aplikacija**-unapred fiksirano, teško promenljivo (npr. rešenja sa TPM ili rana reš. sa Distrib.obj.)
- **Dinamičko partitioniranje aplikacija**-
mogućnost promene alokacije delova apl. na sloj ili mašinu.

Sledeći korak u dinamičkom partitioniranju apl.
je upotreba **s/w komponenata** ⇒ ActivX, Java Beans

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), ActiveX, Java,...
- ❑ **Primeri:** MS Visual Java++, Symantec Cafe Pro,....

Struktura 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, Sybase, Oracle,

Omogućava alatu da podatke dobije na nekoliko načina:

- Podacima se može pristupiti preko **nativnih aplikacionih objekata**, tako rade Smalltalk alati, kao i VB-ovi DAO (danas ADO) objekti.
- Podacima se može pristupiti preko **relacione šeme**, tj. preko tabela, vrsta i kolona.
- Podacima se može pristupiti preko **nativnog drajver interfejsa** kroz pristupni sloj, ili direktno preko **nativnog DB API-a**, zaobilazeći DBA(DB-MW). Na taj način je moguće pozvati servise koje nije moguće pozvati na drugi način.

Repozitorijum-generalno

Termin korišćen sa različitim značenjem i obuhvatom.

- 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.)**
- **OO repozitorijum** predstavlja okvir za definisanje, upravljanje i ponovnu upotrebu različitog oo s/w i komponenata, kao i tradicionalnog s/w i poslovnih komponenti.

Sloj repozitorijuma

- Predstavlja samo još jedan sloj apstrakcije iznad/preko DB sloja.
- Repozitorijum omogućava da se pamti info. o podacima u BP, kao što su: poslovna pravila, ili boja i font atributa koja će se koristiti u celoj apl. kao i definisano ponašanje za slučaj npr. dodavanja vrste u tabelu.
- Neki alati koriste sofisticirane repozitorijume, npr. Power Builder, Uniface, Oracle Developer/2000, a neki ne, npr. Delphi, Visual Basic (VB 5.0 uključuje repozitorijum).
- Neki alati u repoz. pamte i interfejs objekte (JAM 7).
- Obuhvat i kapacitet repozitorijuma veoma variraju od alata do alata.
- Repozitorijum može biti na klijentu, ili može biti na ciljnoj BP, što omogućava da bude lakše korišćen od strane većeg broja programera.

Sloj za dizajn korisničkog interfejsa

- Podsystem alata koji **kreira ekrane, prozore i menije** sa kojima korisnik interaguje.
- Značajno se razlikuju od alata do alata, ali većina obezbeđuje **paletu ili formu** sa koje se može početi rad, kao i **skup kontrola** koje se mogu "dovući" na formu.
- Obično je raspoloživa i **f-ja testiranja**, kako bi se videlo kako interfejs inicijalno reaguje.

Sloj za programiranje

- Podsystem razvojnog alata koji omogućava da se prilagodi ponašanje aplikacije putem upotrebe program. jezika.
- Većina alata su event-driven.
- Programeri smeštaju kod iza kontrola koristeći **editore koda** koji su dostupni sa sloja interfejsa.

Tip koda varira:

- Power Builder koristi Power Script (4GL nalik na Basic), VB koristi VBA.
- Neki alati koriste 3GL (npr. Delphi koristi Object Pascal).
- Ovaj sloj obezbeđuje i debugovanje na nivou koda.

Sloj za deployment (instalaciju)

- Ovaj sloj sadrži sredstva putem kojih se aplikacija prevodi u nešto što se može izvršavati na klijentu.
- Tipično alat kreira pcode i obezbeđuje **runtime interpreter**, oba postoje na klijentu kako bi se apl. mogla izvršavati.
- **Tendncija prelaska na kompajlere**, inic. samo Delphi imao, Power Builder i VB kasnije.
- Ovaj sloj je odgovoran i za isporuku ispravnog DBA sloja, kao i za generisanje apl. objekata koji će se koristiti na AS i DB serv.

Izbor alata

- Uvek delikatno (>200), cene u širokom rasponu: 1.000-75.000 USD. Tehnike:
 - Analiza literature, komparacija svojstava, bitan izbor kriterijuma poređenja.
 - Praktično probno korišćenje, simulacija, prototip.
 - U poslednjem momentu
 - Bez izbora
- Primer:

Izbor alata-model

Diskriminacija alata na bazi 5 kriterijuma, **obeležija modela**:

- ☐ **Poreklo alata**
- ☐ **Tip distribuirane tehnologije**
- ☐ **Obuhvat**
- ☐ **Model komponenti**
- ☐ **Raspon**

Poreklo alata

Važno je znati **poreklo alata**: **MF, supermini računari, PC LAN**

- Oni iz MF okvira su **CASE centrični**.
- Oni iz PC LAN domena su **GUI centrični** i **event-driven** fokusirani.
- Negde u sredini su **4 GL alati** asocirani sa super-mini računarima.
- Pristup koji alat koristi za razvoj apl. je tesno povezan sa njegovim poreklom.

Tip distribuirane tehnologije

Kakve aplikacije kreira alat?

Postoji veza između načina particioniranja sistema i tipa distribuiranog sistema: debeli klijent, debeli server, 3-slojna

- ☐ **Debeli klijent** obično za DSS sisteme
- ☐ **Debeli server** obično za OLTP sisteme
- ☐ **U sredini (npr.3-sloj)** za **Groupware, Distribuirane objekte ili Web**.

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).
- ☐ Neki alati pakuju run-time koji uključuje: TPM, ORB, AS, system mngm, menadžere resursa, kao i jednu tačku instalacije.

Model komponenti

Koji komponentni model alat podržava?

Koje tipove komponenti ima u paleti?

Koje **component foundation libraries** obezbeđuje?

Ranije mnogi alati imali svoj, proizvođački komponentni model. **Danas** alat mora podržavati **de-facto** komponentne **standarde**.

Na strani **klijenta**:

- ☐ Većina alata za debele **klijente** u svojoj paleti podržava ActivX.
- ☐ Internet **klijent** alati podržavaju Java Beans.

Na strani **servera**:

- ☐ Većina alata podržava Enterprise Java Beans (EJB) i CORBA Beans.
- ☐ Neki MS COM+ , Neki mešano.

Raspon-skalabilnost

Koliko 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 preduzeće)

Zahtevi u pogledu idealnog alata za razvoj 3-slojnih c/s aplikacija za objektni web

- ☐ **Bilderi vizualnih mesta za 1-sloj**
- ☐ **Bilder poslovnih objekata za 2-sloj**
- ☐ **Okviri koji enkapsuliraju 3-ći sloj**
- ☐ **Vizualno okruženje za povezivanje komponenata**
- ☐ **Repozitorijum komponenata**

Bilderi vizualnih mesta za 1-sloj

Koji treba da omoguće gradnju **vizualnih kontejnera**, koji se potom mogu napuniti **komponentama**, kao što su:

- **ActiveX**
- **Java Beans**
- **Java Appleti**

Kontejner je : XML compound document obična web stranica

- ☐ Alat mora **obezbediti** prefabrikovane **kontejnere** ili **mesta** (vizuelna tvorevina povezanih komponenti).
- ☐ Ta mesta moraju biti **portabilna** (razne platforme).
- ☐ Moraju biti **”shippable”**, **isporučiva** preko mreže, mora imati priključke (hooks)
- ☐ Alat mora zapakovati mesta u sertifikovane **JAR**-ove sa ciljem distribucije.
- ☐ Brojni **Java alati** danas to omogućavaju
- ☐ **Isporučivo mesto**: mobilni kontejner sa komponentama, tj, mesto koje se može isporučiti preko mreže.
- ☐ Isporučivo mesto omogućava korisniku da interaguje sa višestrukim mestima, kao da se imaju višestruki desk-topovi.
- ☐ Mesto kao virtuelni svet koga klijentima isporučuje server.
- ☐ Mesta omogućavaju serverima da automatski osvežavaju dsktop svojih klijenata.

Bilder 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+ paketiKoji sadrže komponente i njihove klase za podršku

Okviri koji enkapsuliraju 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.

Vizualno okruženje za povezivanje komponenata

- ☐ Moraju obezbediti **vizuelna sredstva za povezivanje (asembliranje) objekata** unutar i između slojeva.
- ☐ Npr. mora postojati mogućnost da se asocira klijentski događaj na 1-sloju sa CORBA ili COM+ pozivom metoda u poslovnom objektu na 2-sloju.

Repozitorijum komponenata

- ☐ Mora podržati tim programera obezbeđujući deljeni **repozitorijum (reusable) komponenata, mesta i serverske objekte**.
- ☐ Treba da omogućiti i **run-time sredstva** koja omogućavaju da se poslovni objekti raspodele na razne tačke (klijent, server, između servera).

Značajni isporučiooci alata

- Inprise/Borland ⇒ Delphi ver.6. , Kylix (Linux)
- Symantec ⇒ Cafe Pro
- Microsoft ⇒ VB ver.6, Visual C++, Visual Studio, Visual Studio.Net
- Oracle ⇒ Designer, Developer 2000,...
- Oracle10g ⇒ iDS-Internet Developer Suite
- Sybase ⇒ Power Soft Power Builder ver.8.
- Dynasty ⇒ Dynasty
- Forte/Sun ⇒ Forte
- Penumbra
- IBM ⇒ Visual Age

SREDNJI SLOJ-MIDDLEWARE

Sadržaj

- Namena MW, funkcije, položaj
- Celovite arhitekture
- Tipovi MW: RPC, MOM,...

Namena MW, funkcije, položaj

- 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)

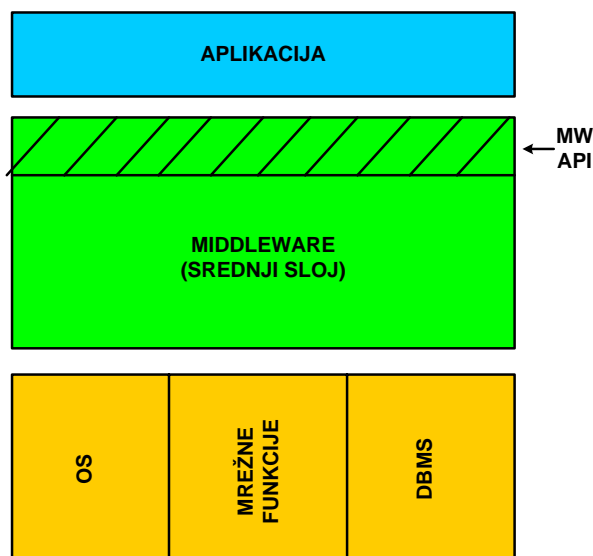
Namena MW, funkcije, položaj

- Upotreba zajedničkog MW API-a "maskira" kompleksnost, kako entiteta (aplikacija ili baza podataka) koji se povezuju, tako i platformi na kojima se oni nalaze.
- Deff: MW je u osnovi bilo koji tip s/w koji omogućava komunikaciju između dva ili više sistema.
- Zbog intenzivnih promena MW je teško kategorizovati.

Namena MW, funkcije, položaj

- Čini 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.
- Aplikativni programeri pristupaju ovim servisima putem MW API-a.

Položaj MW u sistemu



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.

Tipovi MW servisa**Komunikacioni MW:**

- Razmena poruka
- Obrada redova poruka
- Servisi RPC (poziv udaljenih procedura)

Tipovi MW servisa**Informacioni MW:**

- Rečnik (directory)
- Pristup podacima (SQL/RDBMS)
- Deljenje fajlova
- Servisi repozitorijuma
- "Compound" dokumenti

Tipovi MW servisa**Upravljački MW:**

- ORB (Object Request Broker) servisi
- Višestruke niti (multi-threading)
- Servisi upravljanja transakcijama (uključujući TP monitore)

Alternativna podela MW rešenja

- Opšti MW
 - Servisni MW
- ili
- MW nižeg nivoa
 - MW višeg nivoa

Celovite arhitekture⇒alati

Za izgradnju c/s aplikacija u heterogenim distribuiranim okruženjima:

- **OSF DCE** (Open Software Foundation Distributed Computing Environment).
- **OMG CORBA** (Object Management Group Common Object Request Broker Architecture).

Celovite arhitekture⇒alati

- Osnovna sličnost: **MW za C/S**
- Osnovna razlika:
 - **DCE** je bio dizajniran za podršku **proceduralnom** programiranju, dok je
 - **CORBA** bila dizajnirana za podršku **objektno-orijentisanom** programiranju

OSF DCE

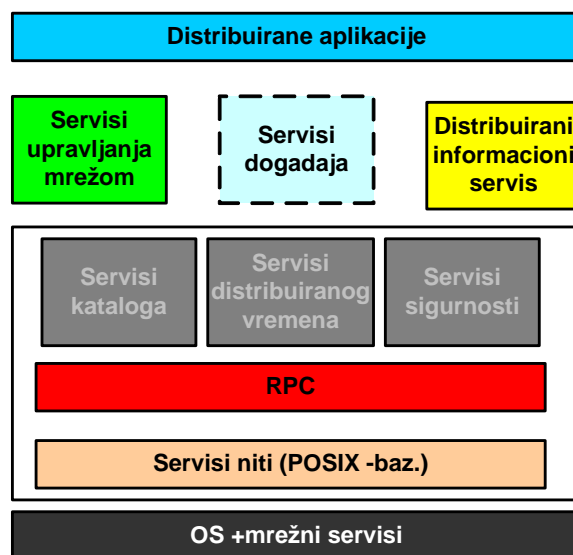
- U osnovi sadrži NOS funkcionalnost, tj. NOS na nivou celog preduzeća.
- DCE je razvijen od strane OSF (danas Open Group), kreira **otvoreno Enterprise NOS okruženje** koje obuhvata višestruke arhitekture, protokole i OS.
- Predstavlja korak na putu razvoja od **Department NOS** (1990), preko **Enterprise NOS** (1996) do **Internet NOS** (1998).

DCE obezbeđuje tehnologije:

- Poziva udaljene procedure (RPC)
- Distribuirane servise imena (Naming services)
- Servise vremenske sinhronizacije
- Distribuiranog fajl sistema
- Servise sigurnosti mreže (security)
- Paket sa nitima (threads)

⇒ **Arhitektura**

OSF DCE Arhitektura



DCE funkcionalnost

- 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 RPC

- Inicijalno sa strane HP-a
- Obezbeđuje **IDL jezik** za def. interfejsa i kompajler.
- **IDL kompajler** kreira prenosivi C kod (stub) za obe strane apl.
- Stub-ovi su kompajlirani i linkovani za **RPC run-time biblioteku**, koja je odgovorna za pronalaženje servera u distrib. sist, obavljanje razmene poruka, pakovanje i raspakivanje parametara poruke, kao i obradu eventualnih grešaka.

Distribuirani naming servisi

- OSF je usvojio ove servise od DEC i Siemens produkata.
- DNS **omogućavaju resursima da budu identifikovani** sa korisnički orijentisanim imenima u namenskoj distribuiranoj BP koja opisuje objekte od interesa.
- **Imena objekata su nezavisna od njihove lokacije u mreži.**

Distribuirani naming servisi

- DCE **servisi direktorijuma** se sastoje od dva elementa:
 - ćelijskog servisa direktorijuma (CDS)
 - globalnog servisa direktorijuma (GDS)
- Obezbeđuje lokalnu naming autonomiju i glob. interoperabilnost.
- Globalni pristup korišćenjem **X.500** naming sistema ili TCP/IP Internet **DNS-a** (Domain Name System).

DCE: Distribuirani servisi vremena

- OSF je ove servise usvojio od Digitala.
- Ova tehnologija **obezbeđuje mehanizme za sinhronizaciju** svakog računara u mreži, sa priznatim **vremenskim standardom**.
- DCE TS obezbeđuje API za manipulisanje vremenskom markom i dobijanje tačnog vremena iz javnog izvora.
- DCE koristi **UTC standard**.

DCE: Distribuirani servisi sigurnosti

- OSF je usvojio sistem autentikacije **Kerberos** (MIT) + HP neka sigurnosna svojstva.
- DCE sigurnosni servisi obezbeđuju: autentikaciju, autorizaciju i upravljanje nalogom korisnika.

DCE: Distribuirani fajl sistem (DFS)

- OSF je izabrao Andrew File System (AFS) od Transarca, kao i deo od HP-a.
- **DCE DFS obezbeđuje uniformni prostor imena, lokacijsku transparentnost fajlova i visoku raspoloživost.**
- Fajlovi i direktorijumi se mogu replicirati na višestruke servere.
- API DFS fajl sistema je baziran na **POSIX 1003.1a** stnd. i interoperabilan sa Sun-ovim NFS-om.

DCE tredovi (niti)

- OSF je izabrao Concert Multithread Architecture (CMA) od Digitala.
- **Tredovi su suštinska komponenta c/s sistema**, koriste se od strane drugih DCE komponenti.
- DCE API je baziran na **POSIX 1003.4a** stnd.
- DCE obezbeđuje i servis semafora koji pomaže redovima da sinhronizuju svoj pristup deljenoj memoriji.

OSF DCE Implementacije:

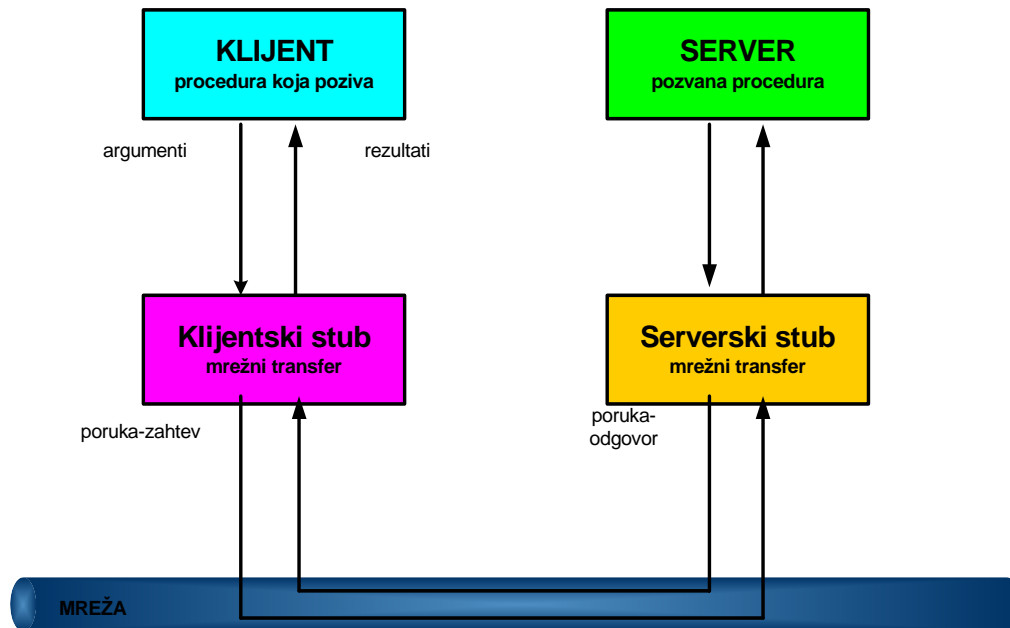
- OSF DCE 1.0 (1992)
- **OSF DCE ver. 1.2.2 (1996)**
- IBM, HP, DEC, Bull, SGI
- **Implementacije na brojnim platformama** (UNIX, Win 2000, Sun Solaris, MVS, ...)
- **Na tržištu nema kompletnog produkta, ali ga ima u drugim:**
- DCE je osnova TPM Encina (Transarc), IBM Directory i security servera,
- Delovi DCE-a, sigurnost i RPC, su inkorporirani u Windows NT 5.0.

Tipovi middleware-a

- RPC (Remote Procedure Call)
- MOM (Message Oriented Middleware)
- DB-orijentisan
- Transakcioni MW (TPM, AS)
- Distribuirani objekti (CORBA, COM/DCOM)
- Brokeri poruka (Message Brokers)

Middleware RPC tipa

- Najstariji tip MW.
- **Omogućava** da se funkcija pozove iz jednog programa, a da se izvršava unutar drugog, na drugom računaru.
- **RPC koristi sinhroni kom. mehanizam**, prekida se izvršavanje prog. da bi se izveo RPC⇒**blokirajući MW**⇒veći "overhead" i slabije performanse.
- U čistom obliku slabo skaliraju. **Arhit.**⇒

Middleware RPC tipa**RPC MW**

- **Klijent i server** rade kao dva odvojena procesa. Ovi **proces**i komuniciraju **putem stub-ova**.
- **Stub** je s/w koji obavlja konverziju lokalnih proceduralnih poziva u seriju mrežnih RPC funkcijskih poziva.
- Stub **određuje adresu** serverskog procesa, usmerava i **konvertuje pozivne parametre procedure** iz interne predstave u standardnu (ASN.1).
- Podaci se predaju RPC transportnom protokolu nižeg nivoa, koji šalje poruke (send, receive)

RPC MW

- Stub je **komunikacioni interfejs** koji implementira RPC protokol i specificira kako se poruke kreiraju i razmenjuju.
- **IDL kompajler** je **ključni deo** koji se koristi za generisanje stub-ova, koji se potom linkuju sa programima klijenta.
- Postoji veći broj **komercijalno raspoloživih protokol kompajlera**:
 - HP: Apollo Network Computing System (NCS)
 - Sun: Transport Independent RPC
 - Netwise: RPC Tool

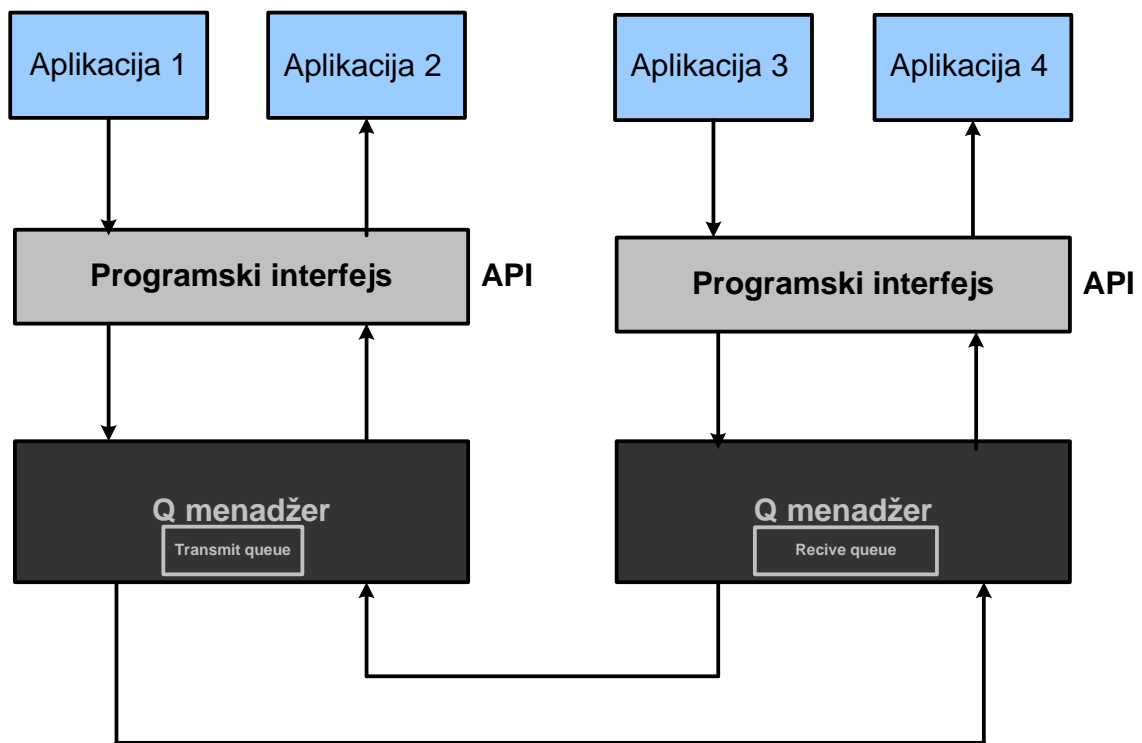
RPC MW

- Većina UNIX sistema isporučuje RPC razvojne biblioteke i alate **kao deo baznog OS**.
- RPC bio osnova OSF DCE, koristi se i od strane MS COM-a, i drugih.
- **Danas IDL kompajler: obično u sastavu OS**
- Zbog ograničenja po performansama **ne preporučuje se korišćenje RPC-a po sporijim mrežama kakva je Internet**.

MW orijentisan na poruke (MOM)

- Kreiran sa ciljem prevazilaženja nedostataka RPC-a, **posredstvom korišćenja poruka (message)**.
- Tradicionalni MOM je tipično **quing s/w koji koristi poruke kao mehanizam za prebacivanje informacije** od tačke do tačke.
- Koristi se **asinhroni** način komunikacije koji ne blokira apl.

Arhitektura ⇒

MW orijentisan na poruke (MOM)**MOM MW**

- ☐ MoM može osigurati isporuku poruke, korišćenjem mehanizma kakav je **persistencija poruke**.
- ☐ **Poruke su male (veličine bajtova)** informacione jedinice koje se kreću između aplikacija, koji imaju strukturu i sadržaj.
- ☐ Moguće koristiti jedan od **dva modela poruka**:
 - -proces-to-proces (oba proc. moraju biti aktivna)
 - -quing model (zgodan za OLTP apl.

MOM MW

- ☐ 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.
- ☐ Obezbeđuje standardne API-e za različite platforme.
- ☐ Dobar izbor za **store-and-forward** kom., kao i za **mreže sa niskom pouzdanošću**.

MOM MW-Primeri produkata

- ☐ IBM: MQ Series
- ☐ Microsoft: MS MQ (Message Que)
- ☐ Java soft: JMS (Java Message Sevice)

Data base orijentisan MW

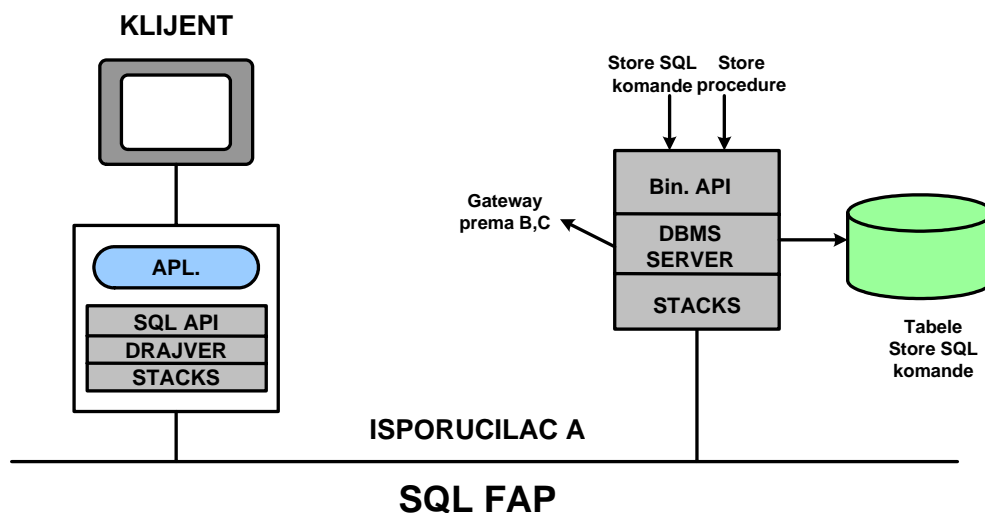
- ☐ U osnovi MW orijentisan na BP.
- ☐ Aplikacioni programi koriste DB MW kao mehanizam za vađenje informacija iz lokalnih ili udaljenih BP.
- ☐ DB MW radi sa nekoliko osnovnih tipova API-a:
 - Native DB-API
 - Call level interface (CLI) API
 - Embedded SQL API

Data base orijentisan MW

- ☐ **Native DB-API:** napravljen za potrebe specifičnog DBMS-a, mogućnost korišćenja svih funkcija.
- ☐ **CLI API:** napravljen kao zajednički interfejs sa ciljem pristupa većem broju različitih BP (RDBMS-ova).
- ☐ **ESQL API:** napravljen sa ciljem direktnog pristupa iz određenog programskog jezika određenoj bazi.

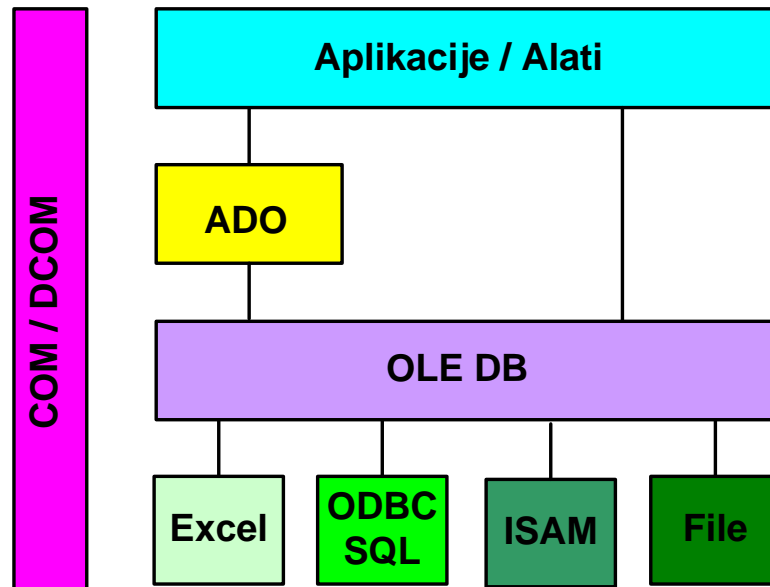
Data base orijentisan MW

- ☐ DB MW u **homogenom** okruženju
- ☐ Jedinstveni protokol: **SQL FAP** (Format and Protocol).
- ☐ Korišćeno DB MW rešenje ima više slojeva:
 - **Odgovarajući MW API**
 - **Drajver za konkretnu bazu (RDBMS)**
 - **Komunikacioni protokol stack**

Data base orijentisan MW

Data base orijentisan MW

- ☐ Konkretna **Microsoft arhitektura**
- ☐ **Interfejs podataka nižeg nivoa OLE DB** obezbeđuje mehanizam za pristup proizvoljnom broju data resursa, uključujući BP, kao standardnim COM objektima.
- ☐ **Interfejs podataka višeg nivoa ADO, ActivX Data Objects (DAO ranije)** obezbeđuje pristup podacima za jezike i alate bez pointera (VB, VBscript, Java script)

Data base orijentisan MW:MS

SREDNJI SLOJ-MIDDLEWARE (2)

Sadržaj

- ☐ Transakcioni middleware (TPM, AS)
- ☐ MW distribuiranih objekata:
 - CORBA
 - COM/DCOM
- ☐ Brokeri poruka (Message Brokers)

1. Transakcioni middleware

Postoji u dve osnovne kategorije:

- Transakcioni monitori (TPM) (**T**ransaction **P**rocessing **M**onitor)
- Aplikacioni serveri (AS) (**A**pplication **S**erver)

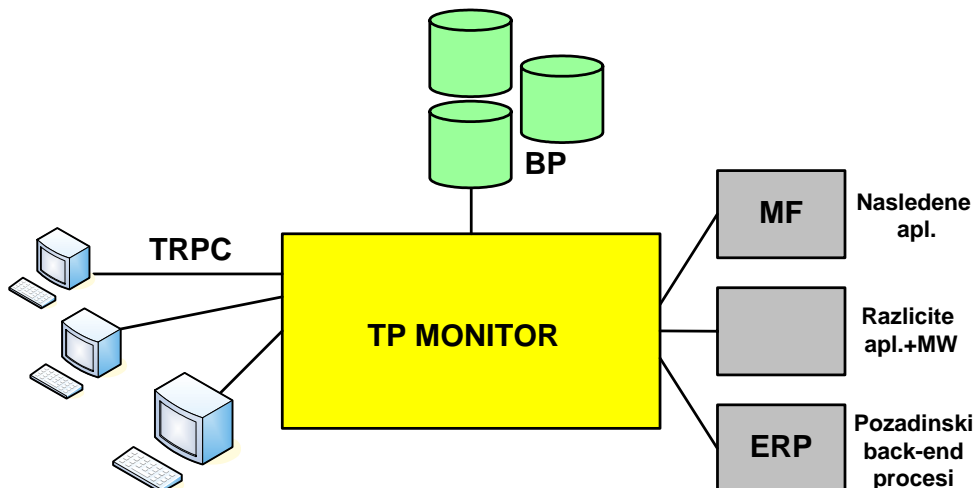
Transakcioni middleware

- Obavljaju posao na koordinaciji transfera informacija i deljenja metoda između brojnih različitih resursa.
- Ova paradigma obezbeđuje odličan mehanizam deljenja metoda.
- Transakcioni MW tipično kreira čvrsto povezano EAI rešenje.

Transakcioni middleware

- ☐ Tr MW obrađuje transakcije u ime klijenta i obezbeđuje:
 - Multipleksiranje BP
 - Balansiranje opterećenja
 - Imunost na greške (fault tolerance)
 - Komunikaciju
- ☐ Klijentska apl komunicira direktno sa TPM putem TRPC, ponekad RMI, IIOP.

Transakcioni MW: TP monitori



Transakcioni middleware

- ❑ Standardi koji definišu kako radi transakcioni **MW** su **DTP (Distributed Transaction Process)** specifikacije donete od strane ISO i X/Open.
- ❑ X/Open 1991. kreirao **TP referentni model**, a 1994. Referentni model distribuiranih transakcija.
- ❑ Jedan od rezultata: XA interfejs koji definiše kako komuniciraju transakc. menadžer i menadžer resursa (DB).

Transakcioni MW: TP monitori

- ❑ Su u stvari transakcioni MW produkti i aplikacioni serveri prve generacije.
- ❑ Obezbeđuju "lokaciju" za aplikativnu logiku osim mehanizma koji omogućava komunikaciju između dve ili više aplikacija.
- ❑ Bazirani su na premisi transakcije, i obezbeđuju konektore niskog nivoa koji omogućavaju da se priključe na resurse (BP, druge aplikacije, que-ovi).
- ❑ Neprevaziđeni su kada se zahteva podrška mnogim klijentima i visoko opterećenje transakcione obrade.

Transakcioni MW: TP monitori

- ❑ Koriste mehanizme: redova ulaznih bafera, kao zaštite od špiceva opterećenja, planiranje prioriteta kako bi se dao prioritet porukama, podrška serverskim nitima kao i mehanizme balansiranja opterećenja.
- ❑ TPM obezbeđuju svojstva: queing-a, rutiranja i slanja poruka, čime se omogućava zaobilazanje TPM-a u slučaju potrebe.

Transakcioni MW: TP monitori

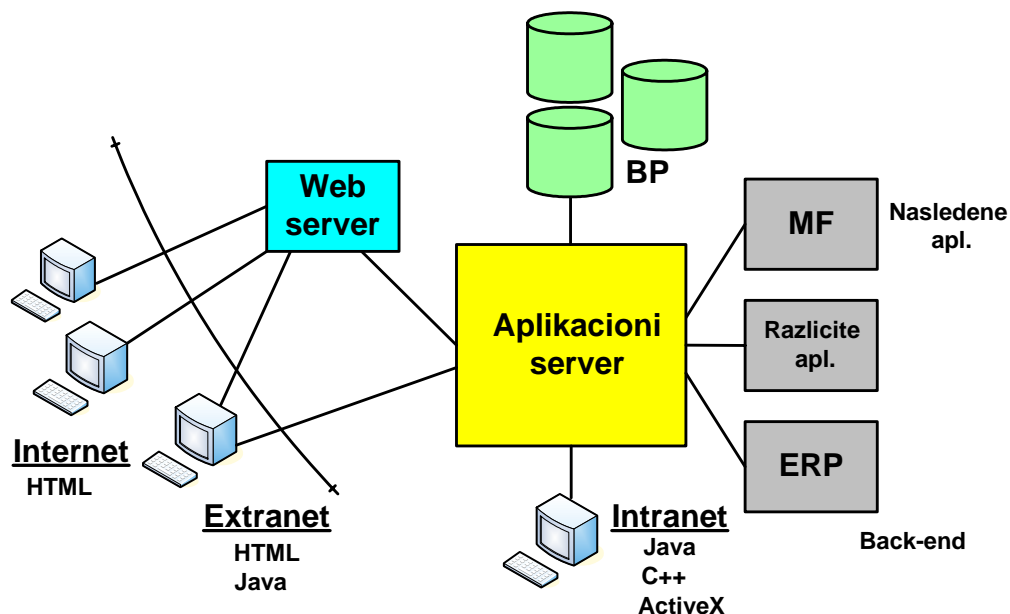
- ❑ **Važno svojstvo balansiranja opterećenja omogućava da se prihvati nalet transakcija, npr. kod periodičnih obrada.**
- ❑ **Primeri:**
 - IBM: CICS (via COBOL) familija
 - Transarc/IBM: Encina (*TXSeries*)
 - BEA Systems: Tuxedo
 - BEA/NCR: Top End
 - Microsoft: MTS (MS Transaction Services) ⇔ OTM

Transakc. MW: Aplikacioni serveri

- ❑ AS nisu nešto novo.
- ❑ AS obezbeđuju **savremeni pristup u deljenju metoda kao i mehanizam za integraciju.**
- ❑ Većina AS se danas **koristi kao Web-enabled MW**, koji obrađuje transakcije od web-omogućenih aplikacija.
- ❑ Koriste oo jezike (npr. Java) umesto, proceduralnih (npr. C, COBOL), koji se često koriste kod tradicionalnih TPM-a.

Transakc. MW: Aplikacioni serveri

- ❑ AS su serveri koji **omogućuju: ne samo zajedničko korišćenje i obradu apl logike, nego i konekciju sa back-end resursima** (npr. BP, ERP apl. čak i nasleđene MF apl.).
- ❑ Npr. sa AS koji zahteva samo oko 50 konekcija, više od 1.000 klijenata može pristupiti DB serveru.
- ❑ Obezbeđuju **mehanizme za razvoj UI.**
- ❑ Obezbeđuju i **mehanizme za spuštanje aplikacije na platformu za Web.**

Transakc. MW: Aplikacioni serveri**Transakc. MW: Aplikacioni serveri**

- ❑ Primeri (Web) AS:
 - Sun/AOL/Netscape/Kiva: Application Server (NAS)
 - Inprise/Visigenic: Application Server
 - IBM: WebSphere
 - BEA/WebLogic: Tengah
 - Oracle Application Server 4.0
 - Microsoft: IIS
- ❑ MW TPM i AS tipa ima značajnu ulogu u EAI.

TPM vs AS

- ❑ AS još ne mogu konkurisati TPM-ovima u pogledu performansi i pouzdanosti ali imaju neka naprednija svojstva, npr. IDE
- ❑ Postavljajući apl logiku u srednji sloj ona se može bolje kontrolisati.

2. Middleware baziran na distribuiranim objektima

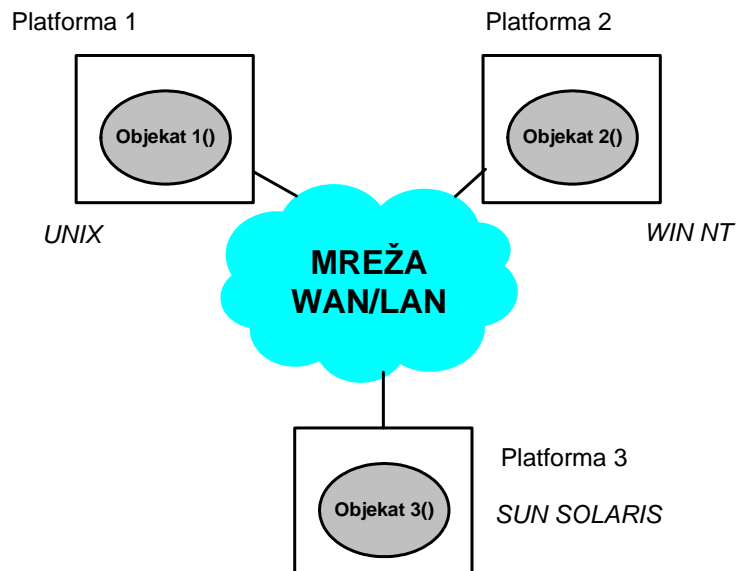
Postoje dve osnovne arhitekture, kao i na njima bazirane tehnologije, alati:

- **OMG CORBA**
- **MS COM/DCOM (Component Object Model)**

MW baziran na distribuiranim objektima

- ❑ Distribuirani objekti su u osnovi mali aplikacioni programi koji koriste standardne interfejsa i protokole za međusobnu komunikaciju.
- ❑ Smatraju se i MW pošto omogućavaju komunikaciju između aplikacija, ali i tehnologiju koja podržava zajedničko korišćenje metoda na nivou preduzeća.

MW baziran na distribuiranim objektima



MW baziran na distribuiranim objektima

- ❑ 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.
- ❑ Omogućeno time što su oba kreirana korišćenjem istog standarda (npr. CORBA) i oba koriste standardni komunikacioni protokol (npr. IIOP).

MW baziran na distribuiranim objektima

- ❑ Distribuirani objekti najbolje rade za **EAI domene koji imaju potrebu da dele veliki broj zajedničkih metoda.**
- ❑ Distribuirani objekti **obezbeđuju potrebnu tehnologiju za deljenje metoda.**
- ❑ Zabluda je da je korišćenje distribuiranih objekata jednostavno, što ilustruje broj propalih projekata!

MW baziran na distribuiranim objektima

- ❑ **Najveća prednost distr. objekata je što se pridržavaju standarda za razvoj aplikacija i interoperabilnost.**

- ❑ **Na tržištu danas postoje dva tipa distribuiranih objekata:**

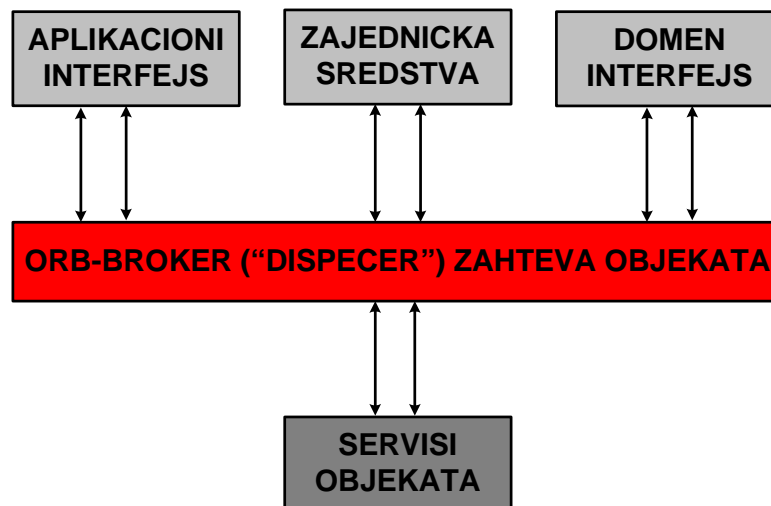
- **CORBA (Common Object Request Broker Architect.)**
- **COM/DCOM (Component Object Model)**

MW baziran na distribuiranim objektima

- ❑ CORBA predstavlja standard a ne tehnologiju. Obezbeđuje specifikacije koje definišu pravila za kreiranje CORBA-usklađenih distr. obj. CORBA je heterogena sa obj. raspoloživim na većini platformi.
- ❑ COM/DCOM je standard za dis. objekte promovisan od strane MS. Obezbeđuje pravila (interfejsi i kom protokoli) za kreiranje COM-omogućenih dis. objekata. Po prirodi COM je homogen, tj. namenjen Windows, mada postoje d.o. i za ne-Win. platforme.

OMG CORBA

- ❑ **OMG konzorijum** osnovan 1989 sa ciljem promocije oo tehnologije, više od 800 članova.
- ❑ **Kreira standarde, specifikacije** (ali ne i s/w) koji **omogućavaju interoperabilnost i portabilnost distribuiranih oo aplikacija**
- ❑ **OMA:** arhitektura upravljanja objektima, sadrži opšti okvir u obliku referentnog modela.

OMG OMA (Object Mngm. Architecture)

OMA (Object Mngm. Architecture)

- ❑ **Sistemske orijentisane komponente:**
 - Servisi objekata (naming, trading, security...)
 - ORB-dispečer zahteva objekata
- ❑ **Aplikativni servisi:**
 - Zajednička sredstva
 - Domen interfejs
 - Aplikacioni interfejs

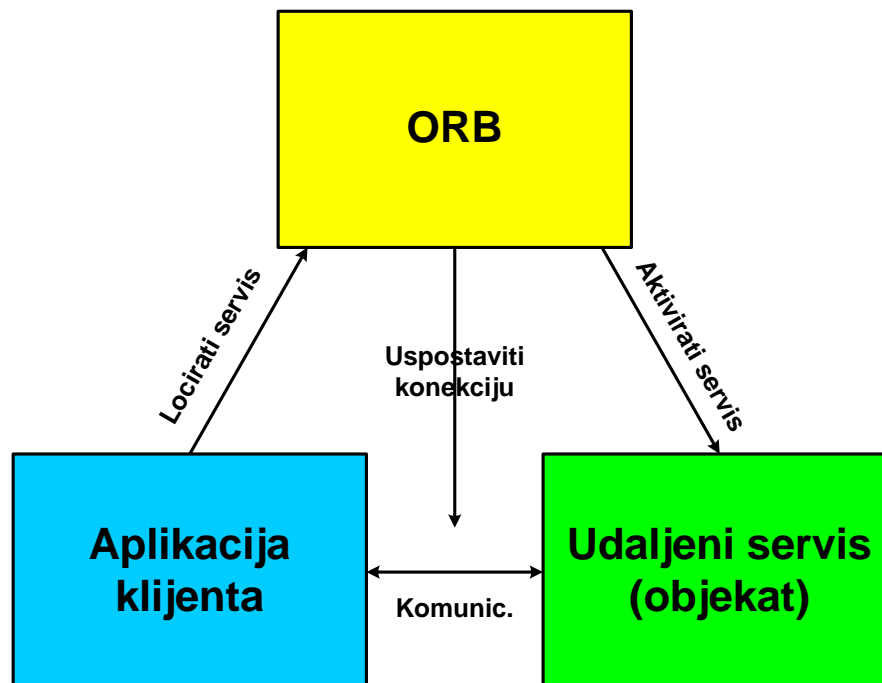
CORBA-evolucija

- ❑ CORBA specifikacija: 1992.
- ❑ CORBA 2.0: 1996, definiše IIOP protokol(Internet Inter-ORB Protokol) koji omogućava interoperabilnost komponenta preko ORB-va različitih isporučilaca.
- ❑ CORBA 3.0: 1998, uključuje i serverske komponente (EJB).

CORBA

- ❑ **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 sarađuju.
- ❑ CORBA je **najznačajniji MW projekat** koji je preduzela IT industrija.

OMG CORBA



CORBA

- ❑ CORBA kao i SQL obezbeđuje statičke i dinamičke interfejsse prema svojim servisima.
- ❑ Posledica alternativnih inicijalnih predloga.
- ❑ Sun +HP: statički API
- ❑ Digital: dinamički API

CORBA

- ❑ Klijentski IDL stub obezbeđuje statičke interfejsse prema servisima objekta.
- ❑ Poziv metoda moguć iz više jezika: C, C++, Java, Smalltalk
- ❑ Statički API se lakše programira, efikasniji
- ❑ Dinamički API teži za programiranje, bolji za alate, fleksibilniji

CORBA-dalji razvoj

- ❑ OMG promoviše **CORBA OTS (Object Transaction Services)**, dok MS putem MTS COM-u dodaje transakcionalnost.
- ❑ **Interoperabilnost se dalje popravlja** sa novim zajedničkim protokolima, npr. CORBA IIOP, MS COM RPC.
- ❑ **Osim sinhronih (RPC) dodaju se i asinhroni mehanizmi razmene poruka**, npr. MS MQ, CORBA putem svog "message" servisa.

MW distrib. obj. :Implementacije

- ❑ Danas su distribuirani objekti postali ključna tehnologija za EAI na nivou metoda.
- ❑ Primeri ORB bazirane implementacije:
 - OMG CORBA: brojne implem. za razne platforme
 - MS COM/DCOM: jedna implementacija, za PC svet (Windows...)
 - Sun Java/RMI: za Java/VM programe

MW distrib. obj. :Implementacije

- ❑ **CORBA/EJB ORB:**
 - Visigenic/Inprise: VisiBroker
 - Iona: Orbix
 - ICL: DAIS
 - BEA: ObjectBroker
 - IBM: SOM
 - JavaSoft: JavaIDL
- ❑ **Microsoft**

MW distrib. obj. : Resursi

CORBA: <http://www.omg.org>

JavaSoft: <http://www.javasoft.com>

MS COM/DCOM:

COM: Knjige o Visual C++ i ActivX

Java i DCOM:

<http://www.microsoft.com/visualj>

<http://www.microsoft.com/java/sdk>

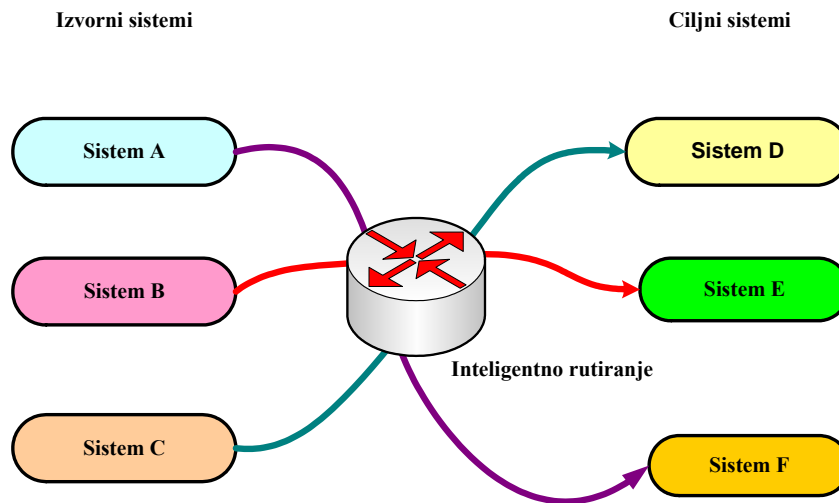
3. Brokeri poruka

(Message Brokers)

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 mogu transformisati šemu (promeniti strukturu poruke) i sadržaj informacije.

Uloga brokera poruka



Brokeri poruka

- ❑ MB predstavljaju (potencijalno) nirvanu za integraciju apl na nivou celog preduzeća (EAI) jer minimiziraju izmene na postojećem sistemu.
- ❑ MB nisu tehnologija za podršku razvoju aplikacija, nego su tehnologija koja omogućava mnogim aplikacijama da komuniciraju jedna sa drugom, tj. oni su "brokeri" informacija.

Brokeri poruka-message brokers

MB obezbeđuju sledeće servise:

- Transformaciju poruka
- Obradu pravila (za obradu i distribuciju poruka)
- Inteligentno rutiranje/Upravljanje tokom
- Repozitorijuma (BP sa info izvornim/ciljnim aplikacijama)
- "Warehousing" poruka (BP poruka)
- Direktorijuma, upravljanja, APIa, Adaptera,
- GUI (kreiranje pravila, def. transform. logike)

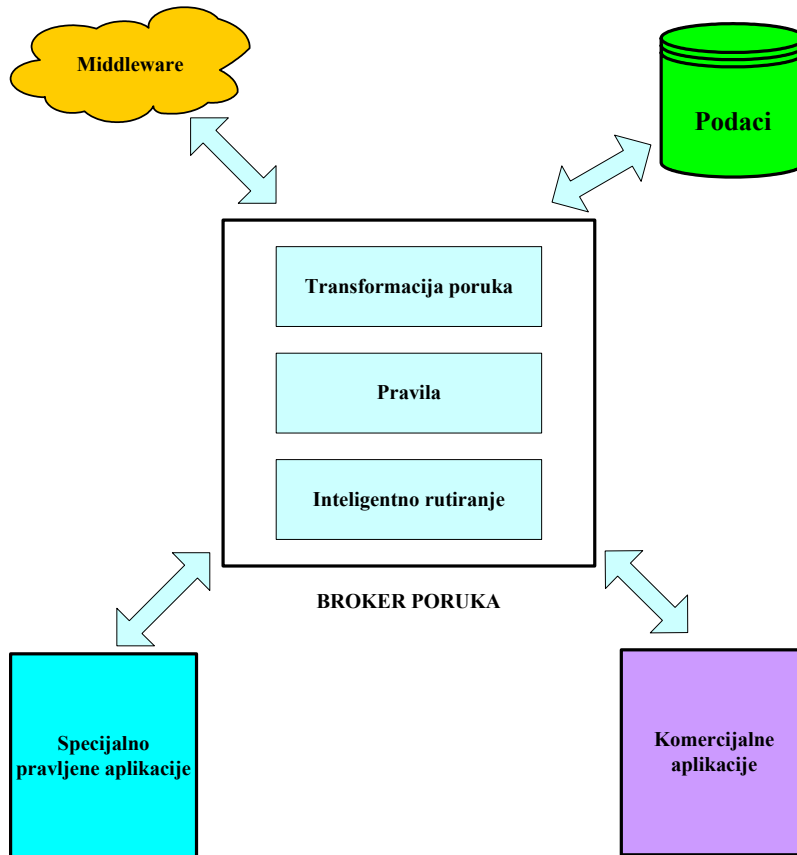
Brokeri poruka-message brokers

MB imaju tri osnovne komponente:

- ☐ Transformacije poruka
- ☐ Mašina pravila (Rules engine)
- ☐ Inteligentno rutiranje

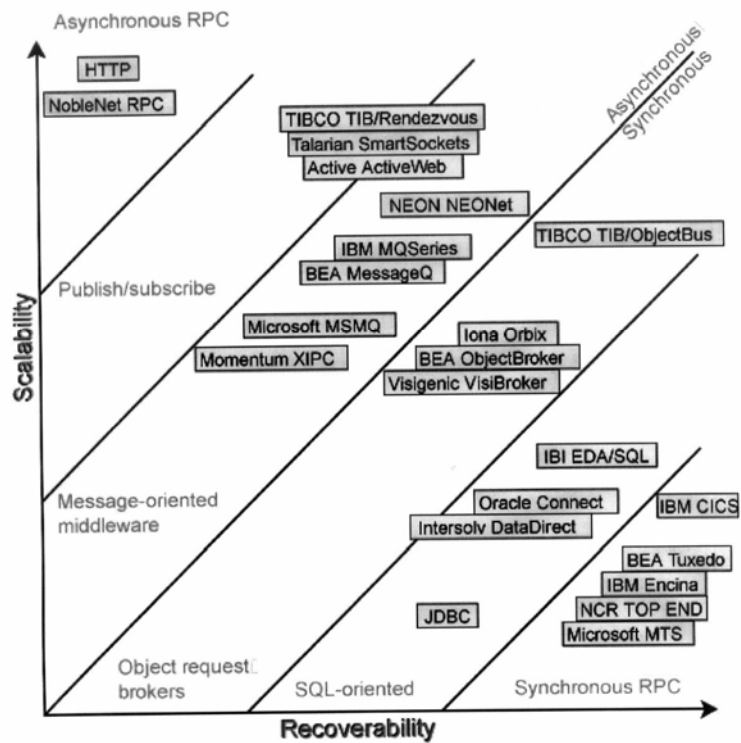
Bazirani su **na asinhronom**, store-and-forward slanju poruka.

MB obezbeđuju servise aplikacijama putem **API-a i adaptera**.

Uloga brokera poruka**Brokeri poruka-message brokers**

Produkti MW MB tipa:

- ☐ NEON (New Era of Networks): MQ Integrator
- ☐ TSI Software: Mercator
- ☐ SAGA software: Sagavista
- ☐ Active Software:
- ☐ Oracle:
- ☐ Microsoft:
- ☐ BEA:

MW: komparacija produkata po tipu i svojstvima

SREDNJI SLOJ-MIDDLEWARE (3)

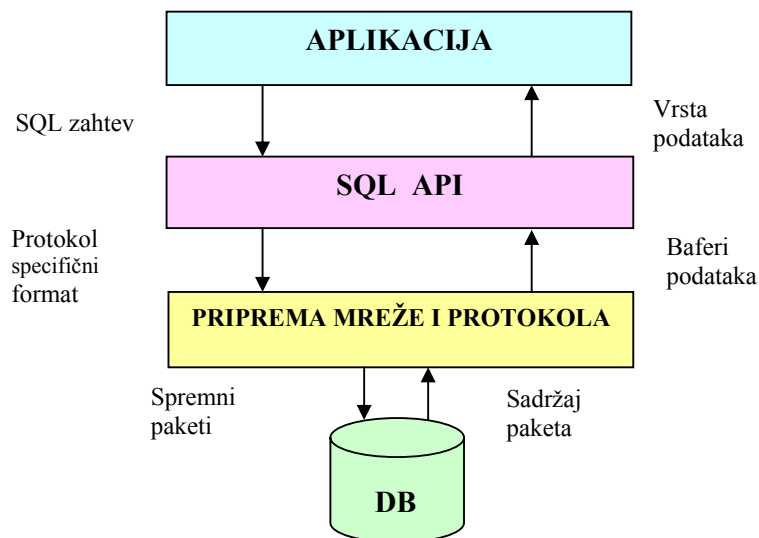
SADRZAJ

- **DB MW** (Data Base Middleware) Srednji sloj za povezivanje sa BP
- Načini (otvorenog) povezivanja klijenta sa DB serverom
- Aplikabilni standardi
- Primeri implementacije/produkata

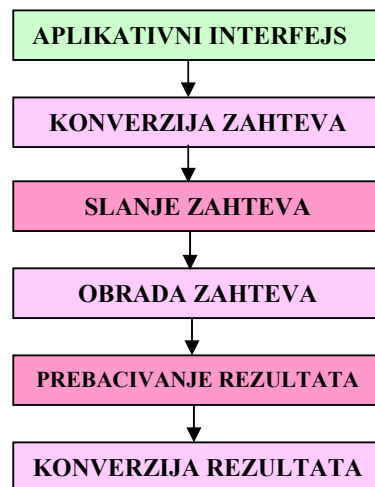
Database orijentisan MW

- 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**.

Pristup BP putem DBA



Generičke funkcije DB MW



Funkcije DB MW

- **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.

Nacini povezivanja klijenta i servera

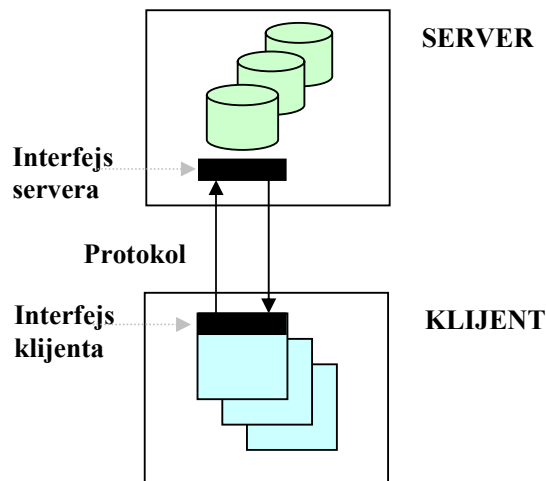
Tipovi DB MW:

- **Nativni (proprietary) DB MW**
- **DB MW tipa zajedničkog interfejsa**
- **DB MW tipa zajedničkog "gateway"-a**
- **DB MW tipa zajedničkog protokola**

Nativni (proprietary) DB MW

- Kreiran je za **određeni tip BP (DBMS)** (npr. Oracle, Sybase, DB/2, Informix)
- Obezbeđuje **najbolje performanse i pristup specifičnim (nativnim) svojstvima** (npr, store procedure, trigeri).
- Tipično su to API-i koje **obezbeđuje proizvođač RDBMS-a**, u obliku C i/ili C++ biblioteka.
- DB MW u **homogenom** okruženju
- **Korišćeno DB MW resenje ima više slojeva:**
 - Interfejs klijenta
 - Protokol
 - Interfejs servera
- **Jedinstveni protokol: SQL FAP (Format and Protocol).**

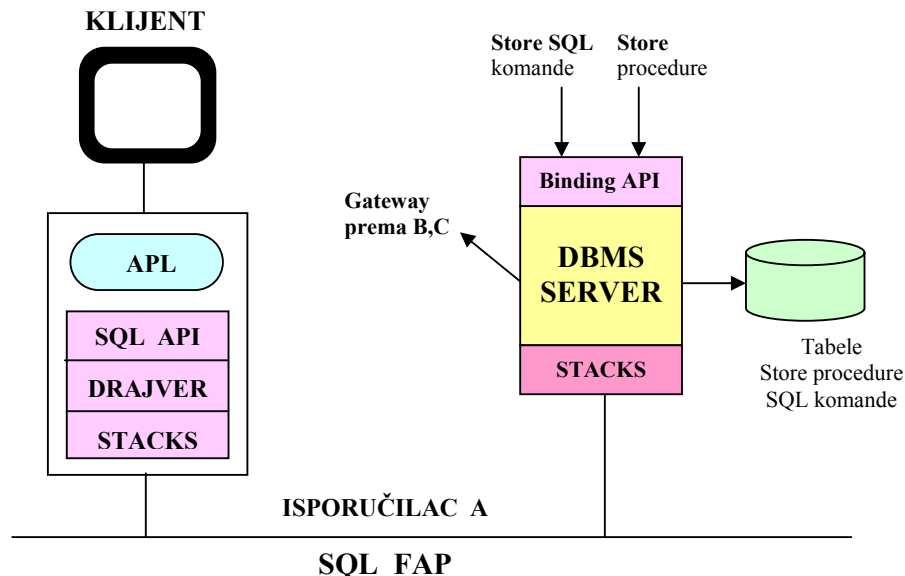
Koncepcija povezivanja c/s sa DB MW u homogenom okruženju



Nativni (proprietary) DB MW

- **Interfejs klijenta**, prihvata zahteve aplikacije i generiše poruku u sklopu protokola, koja se prosleđuje serveru.
- **Interfejs servera**, ima ulogu da prihvati zahteve klijenta i prosledi ih DB engine

Arhitektura →

Arhitektura c/s sistema sa DB MW u homogenom okruženju (native)**Srednji sloj za povezivanje sa BP**

- Tipično MW rešenje za **homogene** uslove danas obezbeđuje:
 - **SQL API**, specifičan za datog isporučioca, koji radi na različitim platformama (**h/w, OS**).
 - **SQL drajver**, specifičan za datog isporučioca
 - **FAP podršku** za veći broj različitih **protokol stekova** (IPX/SPX, TCP/IP,...)

DB MW: nativan, SQL API

- SQL API vecine podrzava **SQL-92** standard.
- SQL API, obicno u dve varijante:
 - **ESQL**-ugnjezdeni SQL API
 - **CLI-SQL API** na nivou poziva

ESQL-ugnjezdeni SQL API:

- dobre performanse
- slaba fleksibilnost
- za ograničeni broj programskih jezika

CLI-SQL API na nivou poziva:

- dobra fleksibilnost
- slabije performanse
- ne zahteva poznavanje ciljne BP

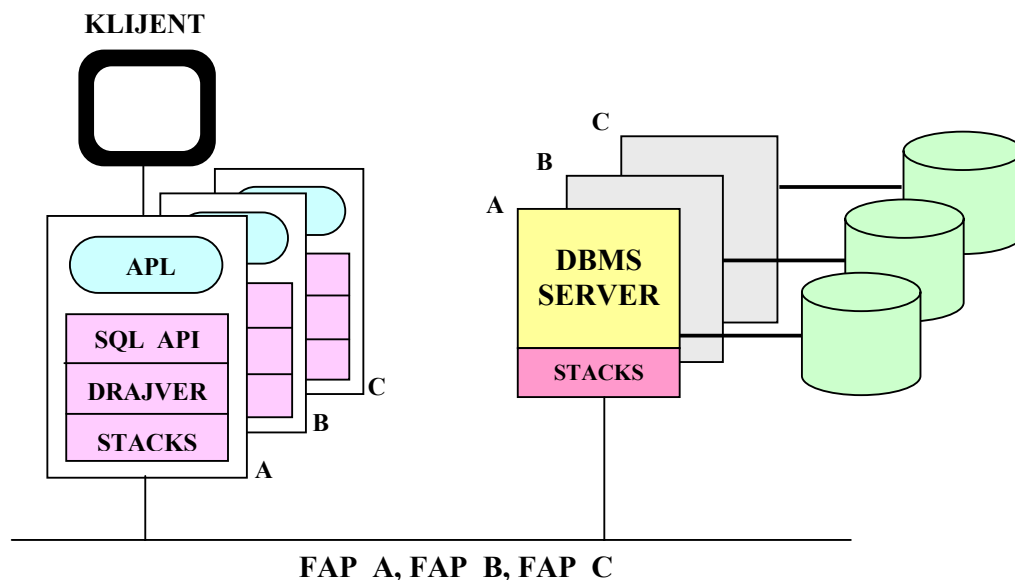
DB MW: nativan, SQL drajver

- To je tipično **run-time element** za klijenta, koji prihvata API pozive, formatira SQL poruku i "rukuje" razmenom podataka sa DB serverom.
- Format SQL poruke i rukovanje istom je poznato kao **FAP (Format and Protocol)**
- **SQL FAP** je tipično definisan od strane isporučioaca (proprietary).

DB MW nativan

Slučaj vise isporucilaca, tj. multi vendor situacija.

Arhitektura →

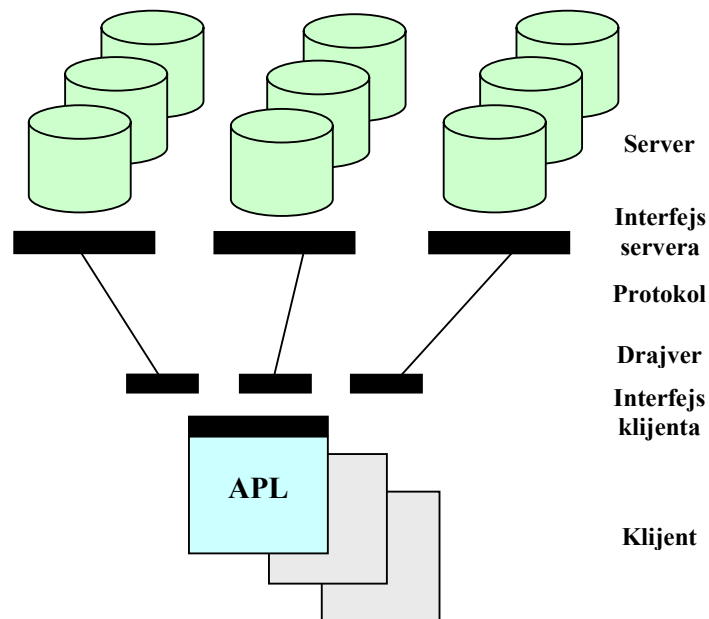
Arhitektura c/s sistema sa DB MW : slučaj vise isporucilaca**DB MW: slučaj vise isporučilaca**

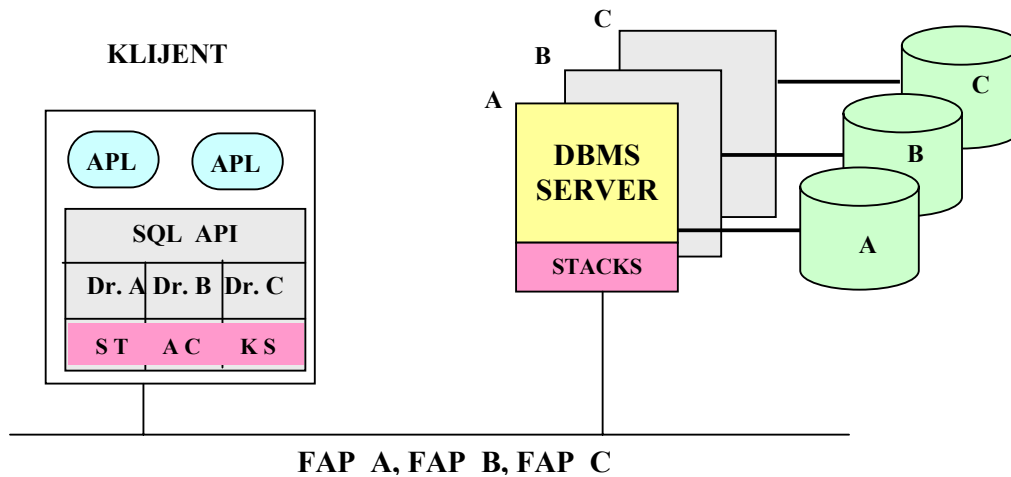
Problemi:

- **Različiti SQL API-ji** bitno otežavaju pisanje zajedničkog skupa aplikacija.
- Čak i za slučaj zajedničkog API-a potreban je način tretiranja proizvođačkih SQL proširenja.
- **Višestruki DB drajveri** troše mem. prostor na klijentu. Kako pratiti koji drajver/verzija na kom klijentu?
- **Višestruki FAP-ovi**, bez interoperabilnosti, impliciraju da različiti DB protokoli samo dele fizičku mrežu, ali ne mogu međusobno sarađivati
- **Višestruki alati za administraciju** impliciraju da DBA mora biti upoznat sa većim brojem upravljački WS, posebnim UI i semantikom.

Povezivanja c/s sa DB MW tipa zajedničkog interfejsa

- Prvi korak u povezivanju je standardizacija zajedničkog SQL interfejsa (SQL API) koji će koristiti sve aplikacije.
- Razlike u serverskoj strani će tretirati drajveri za različite BP.
- **Problemi:**
 - koji SQL API uzeti za standard?
 - Da li ESQL ili CLI API?
 - I dalje trebaju višestruki drajveri
 - Potrebne višestruke ws za upravljanje (DBA), višestruki FAP-ovi

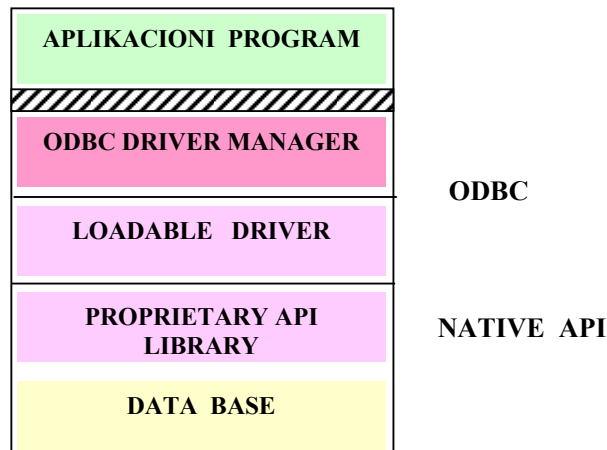
Koncepcija povezivanja c/s sa DB MW tipa zajedničkog interfejsa

Arhitektura c/s sistema sa DB MW tipa zajedničkog interfejsa**Povezivanja c/s sa DB MW tipa zajedničkog interfejsa**

- ESQL standardizovan (ISO SQL-92), dopuna za Javu (JSQL) je deo SQL-3 standarda.
- SQL CLI omogućava kreiranje i izvršavanje SQL instrukcija u run-time-u, u nekoliko varijanata:
 - SAG CLI
 - X/OpenSQLCLI
 - MS ODBC

DB MW tipa ODBC

- **MS Open Data Base Connectivity (ODBC)** Windows std za SQL, proširena verzija SAG CLI-a.
- **MS ODBC 1.0 SDK**: 1992. (DBA za Win), spor, bagovit.
- **MS ODBC 2.0 SDK**: 1994. (za 32 bitne)
- Vecina DB server vendora danas podrzava ODBC API, osim nativnih APIa
- Visigenic: ODBC SDK za ne Win. platforme.
- Intersolv: Open Link
- **Problem**: vecina server vendora dodaje "proprietary" proširenja serv drajverima i **ODBC CLI**.
- **MS ODBC 3.0 SDK**: 1996. (Unicode)
- **MS ODBC 3.5 SDK**: 1998.

Arhitektura DB MW tipa ODBC**DB MW tipa ODBC-nedostaci**

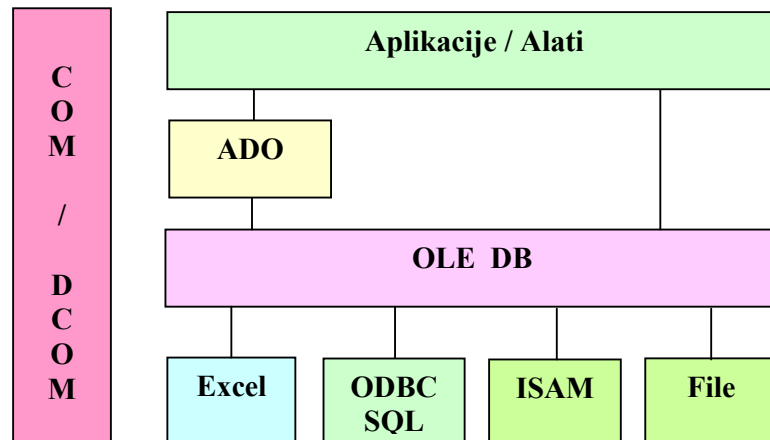
- MS potpuno kontroliše ODBC standard
- Stalno se menja
- Pitanje kada će biti usklađen sa SQL-3
- Nejasna budućnost vs. OLE DB
- Razliciti nivoi usklađenosti
- Značajan overhead
- Obicno sporiji od nativnog API
- OK za read-only funkcije

DB MW tipa 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, rezultujuće skupove, meta-podatke BP i druge DB objekte.
- Korišćenjem JDBC Java apl. i apleti mogu izvorsavati SQL iskaze i memorisane proc. iz relacione BP.
- Postoje **JDBC** drajveri različitog tipa

MS OLE DB i ActiveX Data Objekti

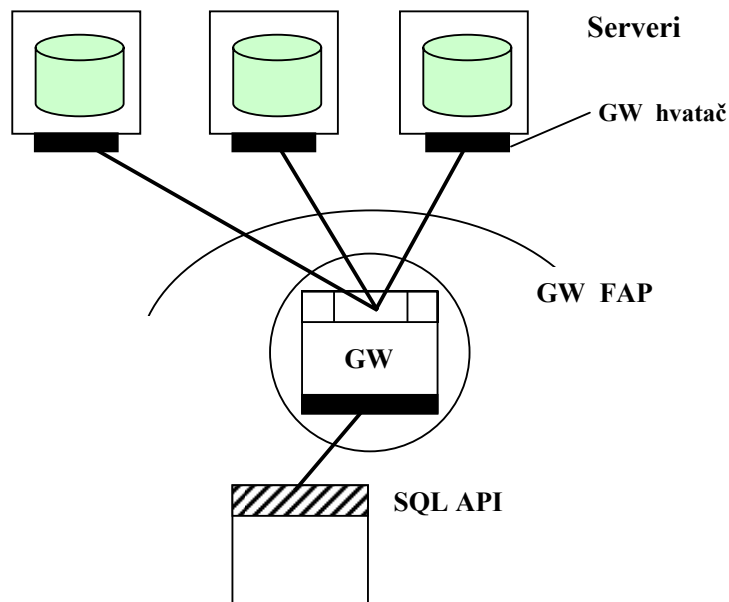
- **OLE DB** je rešenje za pristup podacima za Windows C i C++ programe
- OLE DB obezbeduje **pristup razlicitim izvorima pod** (relacione, vise dim,...)
- **ActivX Data Objekti (ADO)** je objektni sloj preko **OLE DB** koji obezbeduje pristup pod. jezicima bez pointera (Visual Basic) i skript jezicima tipa Java Script i VB Script.

MS arhitektura DB MW: OLE/ADO**DB orijentisan MW: OLE DB**

- Konkretna **Microsoft arhitektura**
- Interfejs podataka nižeg nivoa OLE DB obezbeđuje mehanizam za **pristup** proizvoljnom broju **data resursa**, uključujući BP, kao **standardnim COM objektima**.
- **Interfejs podataka višeg nivoa ADO, ActivX Data Objects (DAO** ranije) obezbeđuje pristup podacima za jezike i alate bez pointera (VB, VBscript, Java script)

Povezivanje c/s sistema DB MW tipa zajedničke kapije (gateway)

- Posle standardizacije SQL API-a moguće je standardizovati jedan (dva) otvorena industrijska FAP-a (**gateway FAP**) i na bazi njega zajednički klijentski drajver (**gateway driver**) za taj FAP.
- **gateway "hvatač"** za svaki od servera koji hvata dolazeće FAP poruke i prevodi ih prema nativnom SQL interfejsu lokalnog servera.



Povezivanje c/s sistema DB MW tipa zajedničke kapije (gateway)

- Danas postoje dva "zajednička" FAP-a :
 - IBM: **DRDA**
 - IB: **EDA/SQL (Enterprise Data Access)** ali koji nisu još standardizovani.

Primeri (stariji):

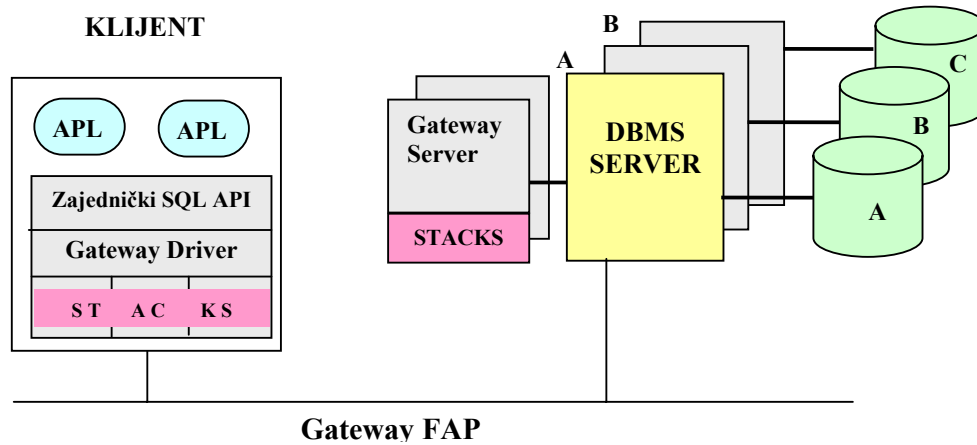
MDI: DB gateway za DB2

Sybase: Open server

IBM: DDCS/2

MS: ODS

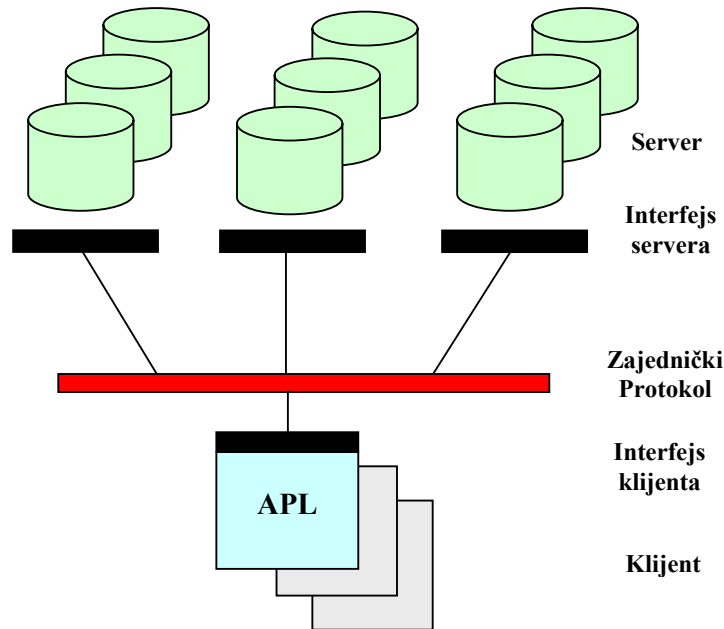
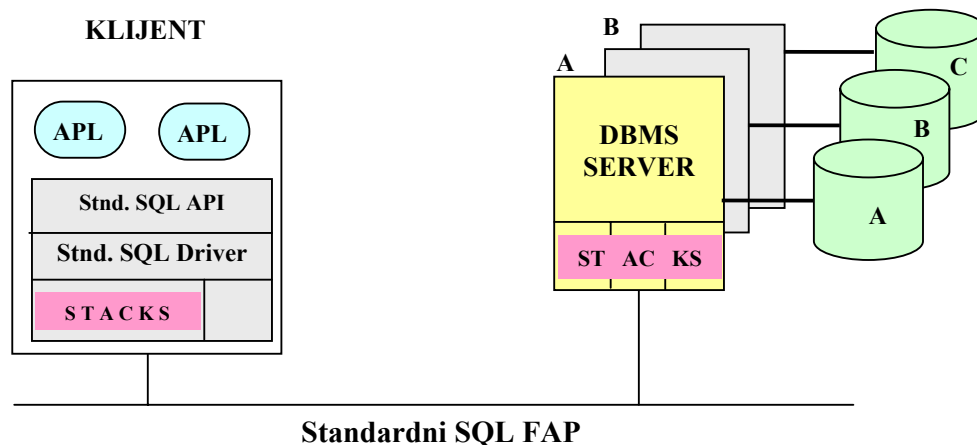
Arhitektura c/s sistema sa DB MW tipa zajedničkog gateway-a



Povezivanje c/s sistema DB MW tipa zajedničkog protokola

Nirvana! Koja podrazumeva;

- Standardni, zajednički SQL API
- Standardni SQL driver
- Standardni zajednički SQL FAP

Koncepcija povezivanja c/s sa DB MW tipa zajedničkog protokola**Arhitektura c/s sistema sa DB MW tipa zajedničkog protokola****Povezivanje c/s sistema DB MW tipa zajedničkog protokola**

- Time se **eliminišu GW hvatači** (cena, održavanje, performanse)
 - Podrazumeva se da **zajednički FAP mora podržavati ili super-set svih SQL dijalekata** ili **DB vendori moraju zameniti svoje FAP-ove standardnim!**
 - Posebno pitanje administracija u heterogenom okruženju.
- Standard postoji: ISO RDA, ali ...**

