Uvod u distribuirane informacijske sustave

Procesi u klijent server komunikaciji

Da ponovimo …

- Proces
  - IPC
- Nit
  - User level
  - Kernel level
  - Leight weight processes

Sadržaj

- Virtualizacija
- Procesi klijenta
- Procesi servera
- Migracija koda
- Komunikacija

Virtualizacija

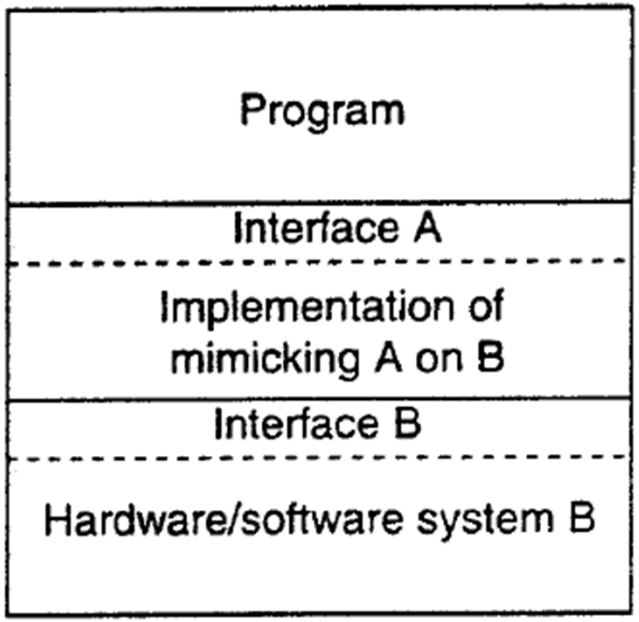
- Niti i procesi daju privid postojanja više resursa (više procesora)
- Ovaj se pristup može proširiti i na ostale resurse – virtualizacija resursa
- Brzi razvoj softwarea i hardwarea – aplikacija gotovo uvijek nadživi operacijski sustav i hardver koji je podržava
  - Stvoriti virtualno okruženje na novom sustavu na kojem će takva aplikacija raditi

Virtualizacija

- Virtualizacija je kreiranje virtualne verzije nečega, poput hardwarske platforme (npr. mobitel) operacijskog sustava (cygwin), uređaja za pohranu (virtualni disk) ili mrežnih resursa
  - Olakšava administraciju velikog broja servera ili resursa
  - Potpomaže skalabilnost i bolje iskorištavanje resurasa
  - Pogon iza cloud computing i utility computing
  - Koristi se odavno (npt 70-tih IBM-ova mainframe računala koriste ovu tehniku)

Uloga virtualizacije u ditribuiranom sustavu

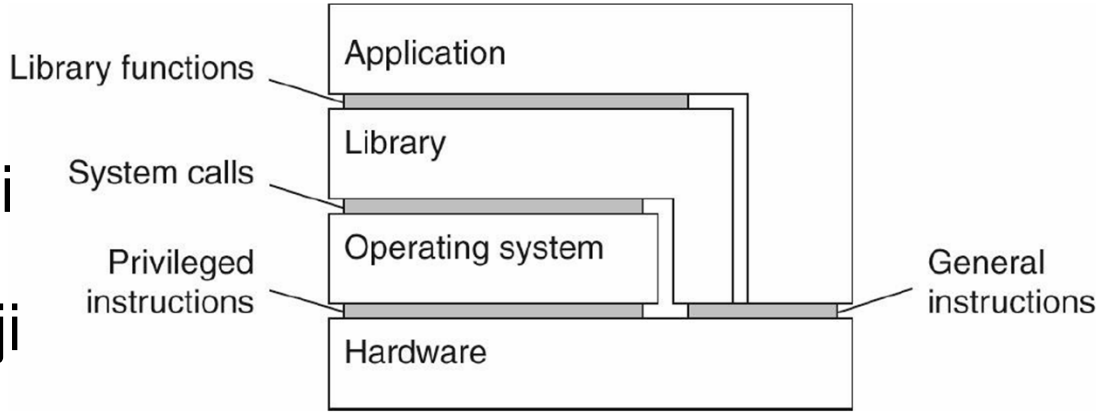
- Virtualizacija se bavi proširivanjem ili zamjenom postojećeg sučelja sa tako da imitira ponašanje nekog drugog sustava



Arhitekture virtualnih mašina

- Razine sučelja:

1. Između hardwarea i softwarea koje sadrži strojne naredbe koje može pozvati bilo koji program



1. Između hardwarea i softwarea koje sadrži strojne naredbe koje može pozvati samo privilegirani program (npr. OS)
2. Sučelje koje se sastoji od sistemskih poziva koje nudi operacijski sustav
3. Sučelje koje se sastoji od poziva biblioteka – (često se i sistemski pozivi skrivaju iza API)

Arhitekture virtualnih mašina

- Virtualizacija se može implementirati na više razina:
  - Process Virtual Machine: abstraktni skup naredbi koji se koristi za izvršavanje aplikacije, npr. Java runtime ili Windows emulator (Wine)
  - *Virtual Machine Monitor*: sloj koji u potpunosti skriva postojeći hardware, ali nudi potpuno novi skup naredbi na istom hardweru kao sučelje. Ovo omogućava imati više operacijskih sustava na jednoj plsatformi npr.: Vmware, VirtualBox, Xen, VirtualPC, Parallels etc.

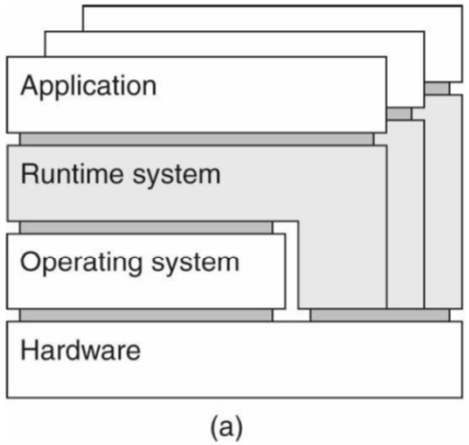


Figure 3-7. (a) A process virtual machine, with multiple instances of (application, runtime) combinations.

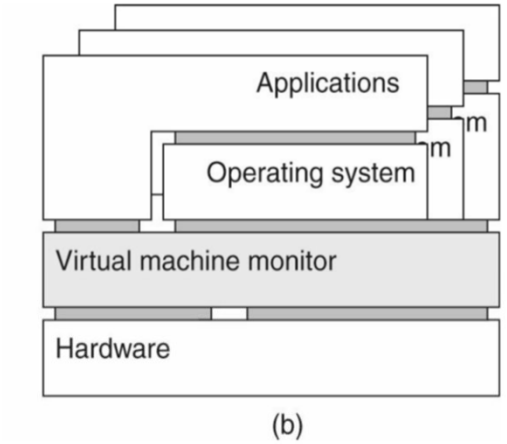


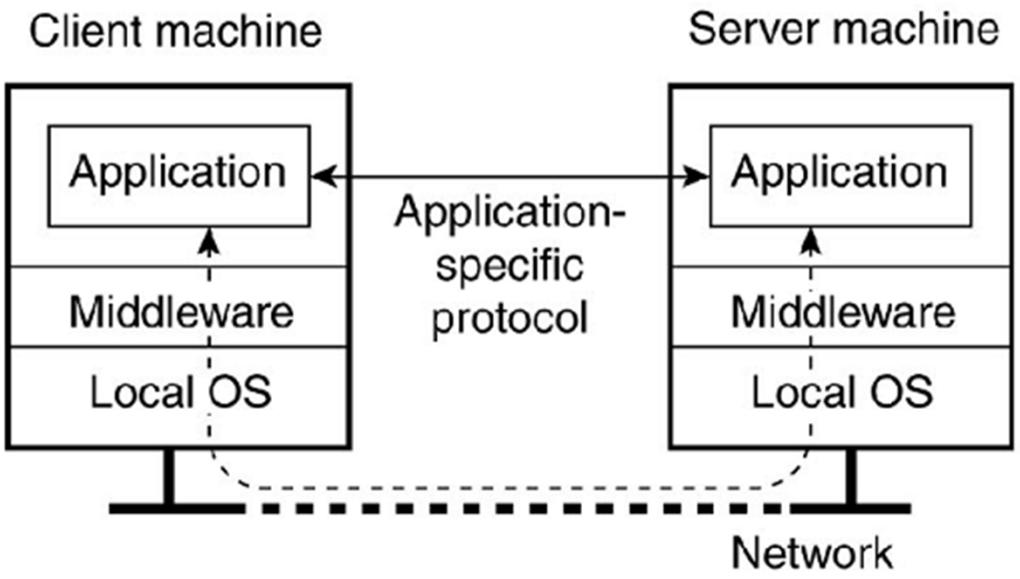
Figure 3-7. (b) A virtual machine monitor, with multiple instances of (applications, operating system) combinations.

Procesi klijenta i procesi servera

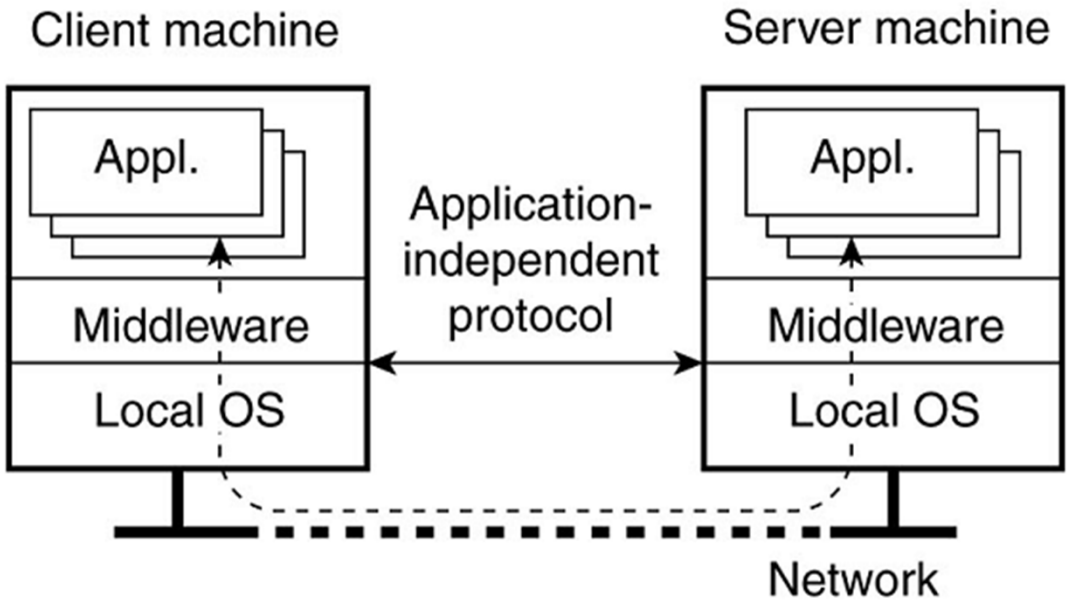
- Klijent server arhitektura igra važnu ulogu u distribuiranim sustavima
- Pogledajmo malo detaljnije anatomiju klijenata i servera

Klijenti

- Mrežna korisnička sučelja
- Sredstvo za komunikaciju sa udaljenim serverom
- Ugrubo dva načina podržavanja komunikacije:
  - Na razini aplikacije: klijent za uslugu kojoj pristupa ima odgovarajući dio kojim pristupa serveru (npr. PDA i raspored)
  - Na raini middlewarea : direktan pristup servisu nudeći samo korisničko sučelje (thin client)



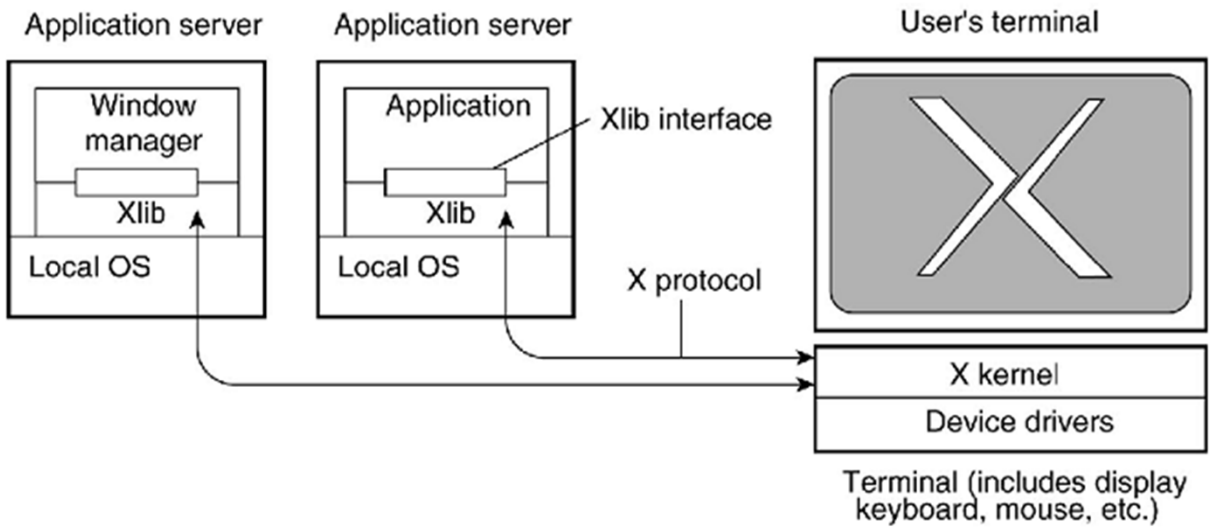
1. Mrežna aplikacija koja koristi svoj protokol



2. Mrežna aplikacija koja koristi generalni protokol

X Window sustav

- Programski sustav i protokol za mrežna računala
- 1984 MIT, trenutna verzija X11(1987)
- Omogućava GUI (Graphical User Interface) i mogućnost upravljanja (rich input device) terminalima za udaljena računala
- Kroz sloj abstrakcije hardwarea
- X kernel – driveri za upravljanje uređajima terminala – upravljanje ekranom i dohvat eventa tipkovnice i miša
- Xlib - koriste aplikacije



- Aktivnost korisnika, korištenjem miša ili tipkovnice na terminalu rezultira aktiviranjem X kernel lokalnih događaja koji šalju zahtjeve Xlib
- Xlib je sučelje udaljene aplikacije koje šalje zahtjev X kernelu kako izmjeniti izgled sučelja
- X kernel može istovremeno komunicirati sa više aplikacija
- Window manager - upravlja izgledom GUI sučelja (raspored prozora, gumbova..)
- Što je klijent, a što server??
  - X kernel prima zahtjeve kako da preuredi izgled ekrana – ima ulogu servera

Thin klijent

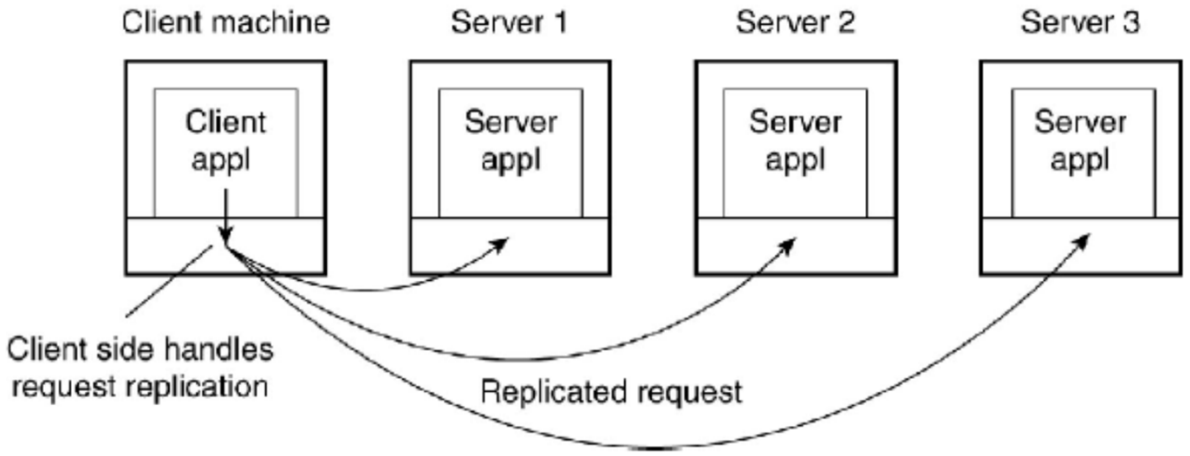
- Aplikacije upravljaju izgledom sučelja korištenjem komandi definiranih X standardom
- Logika aplikacije trebala bi biti izdvojena od korisničkog sučelja
- Nažalost ovo nije uvjek slučaj – često aplikacije traže aktivnost korisnika prije nego nastave sa radom – sinkrono ponašanje
- Dva pristupa rješavanju ovoga:
  - NX: kompresija poruka korištenjem lokalnog cachea
    - djeljeni podaci
  - Upravljanje izgledom ekrana na strani aplikacije na razini piksela – zahtjeva dobru kompresiju podataka
    - Gubi se semantika aplikacije
- THINC – video, queued messages

Složeni dokumenti

- Više aplikacije dijeli jedan GUI prozor
- Drag-and-drop operacije : npr premještanje ikone datoteke u „smeće” = dohvatiti ime datoteke, prosljediti ga aplikaciji povezanoj sa „smećem”
- Editiranje na licu mjesta: npr. Dokument koji se sastoji od teksta i slike
  - ako se korisnikov kursor nalazi iznad teksta omogućiti unos teksta,
  - ako je iznad slike omogućiti obradu slike
- Složeni dokument: sastoji se od teksta, slika, tablica
  - Korisničko sučelje treba sakriti činjenicu da se radi o različitim aplikacijama

Klijentski software za transparentnost distribucije

- Osim korisničkog sučelja, klijentski softer mora skrivati distribuciju
- Replicirani serveri
- Transparentnost pristupa – postiže se generiranjem klijentovog zapisa
- Lokacija, migracija i relokacija: odgovarajućom konvencijom imenovanja
- Klijentska rješenja – klijentov middleware skriva relokaciju, replicira zahtjeve na replicirane servere, ponavlja zahtjeve u slučaju greške

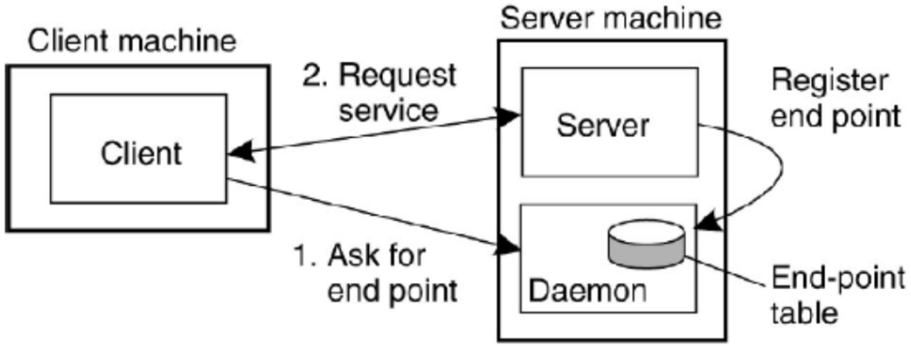


Serveri

- Server nudi specifičnu uslugu skupu klijenata
- Proces koji prima zahtjeve, obrađuje ih te čeka sljedeći zahtjev
- Iterativni server - sam obrađuje zahtjeve
- Konkurentni server: zahtjev prosljeđuje posebnoj niti

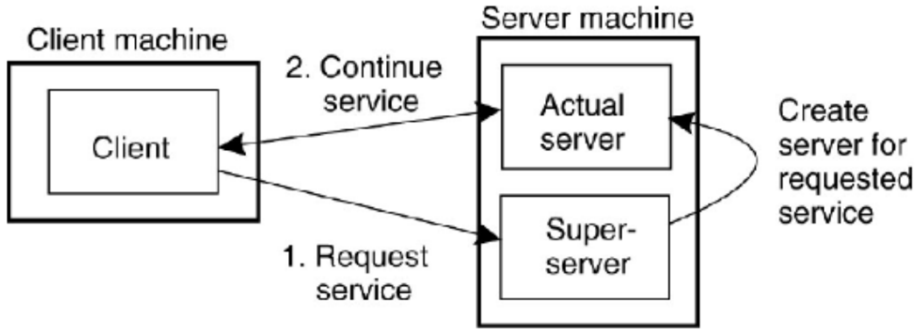
Serveri – portovi

- Koja je krajnja točka kontakta sa serverom: port
- Svaki proces “sluša” na jedinstvenom portu
- Koji port kontaktirati :
  - Standardni broj porta usluge (IANA: FTP-21, ssh-22, WWW -80…)
  - deamon



Serveri – portovi

- Server ima mnogo procesa koji se simultano odvijaju
- Većinu vremena procesi su pasivni, čekaju na zahtjev
- Umjesto praćenja velikog broja procesa -imati jedan superserver
- Inetd sluša veći broj poznatih portova. Kada zaprimi zahtjev pokreće proces koji ga obrađuje. Nakon obrade proces se gasi.



Serveri

- Prekidanje zahtjeva:
  - Najjednostavniji način – prekid komunikacije
  - Slanje out-of-band podatka
  - Otvoriti posebni komunikacijski kanal ili poslati istim kanalom (TCP dozvoljava “urgent” pakete)

Serveri

- Stateless ili stateful
- **Stateless** serveri:
  - ne prati stanje klijenta,
  - svoje stanje može promijeniti bez da obavjesti klijenta
  - Primjer: web server – svaki zahtjev tretira kao zaseban
    - HTTP 1.0 – svaki zahtjev zaseban
    - HTTP 1.1 – session
  - Web server također ima log dokument
- Soft stanje – server pamti podatke o klijentu određeno vrijeme. Nakon isteka vremena zaboravlja ga

Serveri

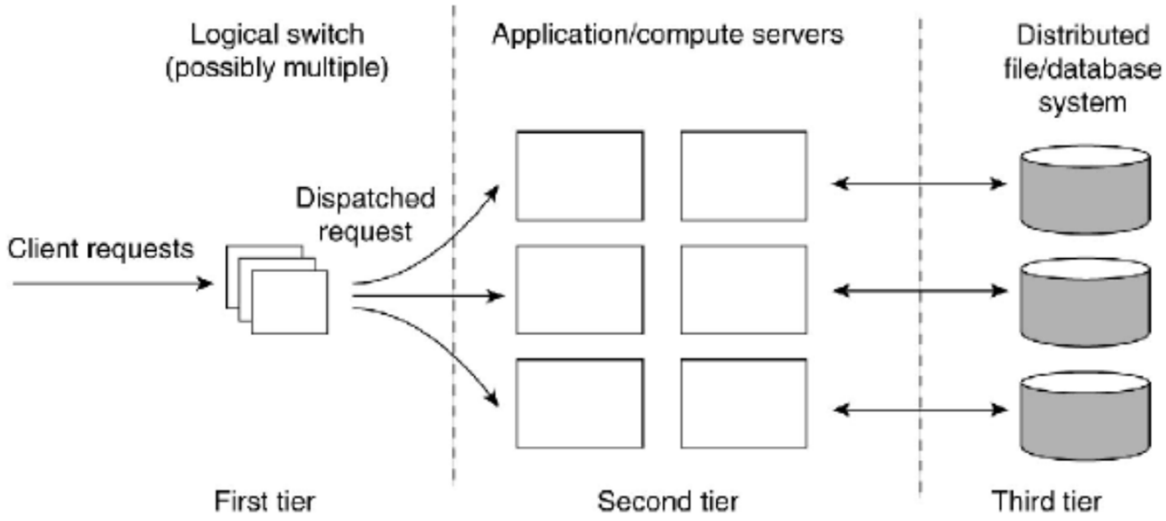
- **Stateful** serveri: čuva informaciju o klijentu sve dok ne primi eksplicitan zahtjev za brisanjem podataka
- Primjer : dokumentni server (file server)
  - Pamti tablicu unosa (klijent, dokument) sve dok klijent ne obriše dokument
  - Također pamti koji klijent ima prava izmjene, prati zadnju izmjenu…
  - U slučaju rušenja servera – tablica se mora obnoviti.
  - Stanja moraju biti trajno pohranjena

Serveri

- Session stanja VS permanentna stanja
- Permanentna stanja nije moguće izmjeniti
- Dizajn servera ne bi trebao utjecati na uslugu
  - Npr. Pisnje i čitanje dokumenta može se izvesti i stateless serverom
  - Čuvanje informacija na klijentovoj strani (cookie)

Klasteri servera

- Klaster je kolekcija umreženih računala povezanih brzom vezom (LAN) koji pogone nekoliko servera
- Obično je organizacija klastera 3-tier arhitektura



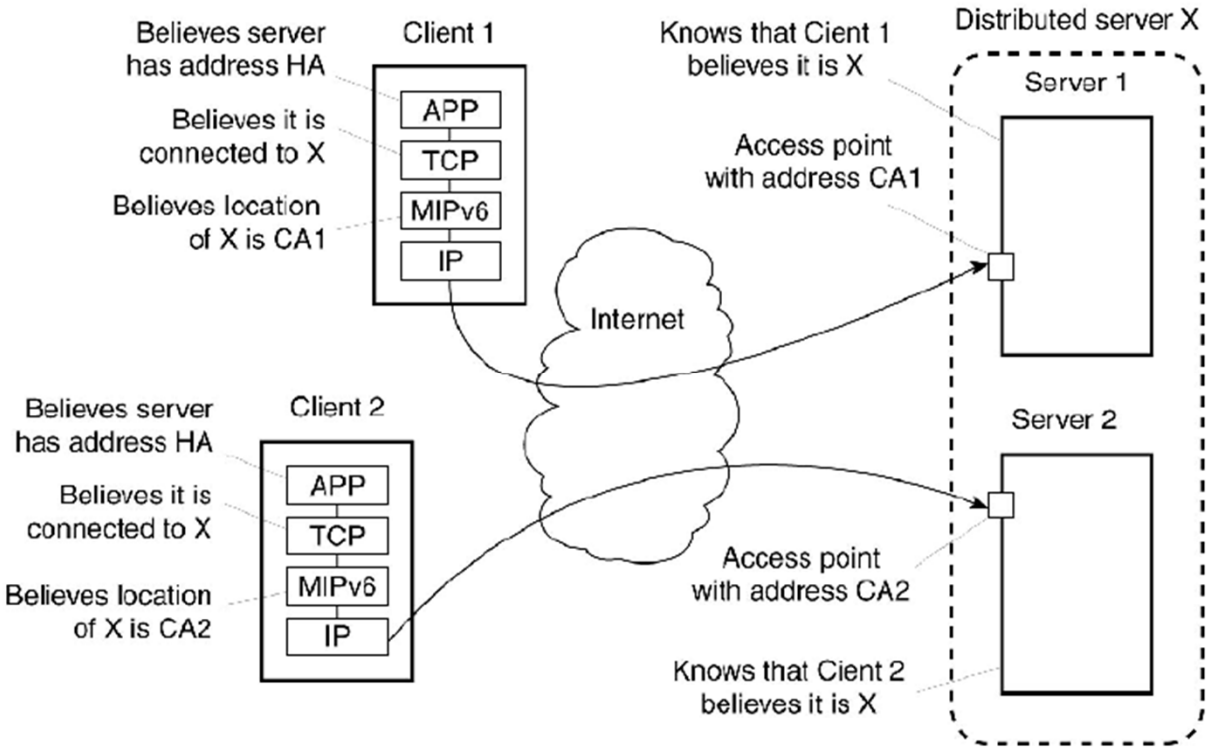
Distribuirani serveri

- Klasteri su statički konfigurirani, imaju jednu krajnju točku – ovisan o funkcioniranju tog računala
- DNS – može za isti “host name” vratiti nekoliko adresa
  - klijentu će iskuša alternativne adrese ako jedna ne uspije
  - No ipak – statičke adrese
- Rješenje – distribuirani server – dinamički promjenjivi skup računala sa varirajućim točkama pristupa
- Ovaj pristup osigurava stabilne servere

Distribuirani serveri

- Koristi se svojstvo mobilnosti iz MIPv6 protokola
  - Svaki mobilni čvor ima home netvork, home addres (HoA) te Home Agenta (HA). Kada se nalazi u drugoj mreži dobije Care-of address (CoA) i obavjesti se HA koji preusmjerava komunikacje na CoA
- Distribuirani server ima jednistvenu kontaktnu adresu početno pridjeljenu klasteru.
- Kontaktnu adresu može prieuzeti bilo koji čvor u slučaju potrebe

Distribuirani serveri

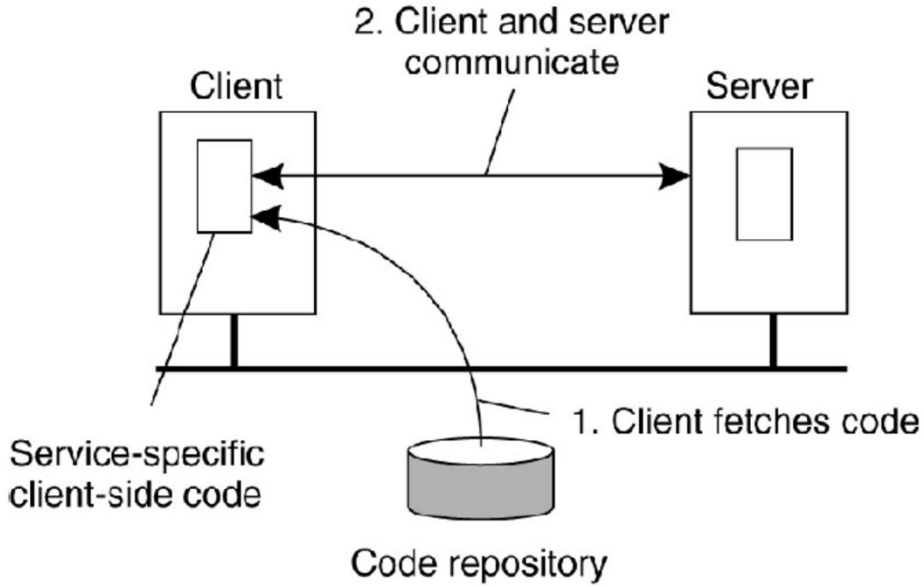


Migracija koda

- Komunikacija – prenošenje podataka
- U nekim situacijama potrebno je prenjetui programe
  - Nekada i za vrijeme izvršavanja
- Migracija procesa: prebacivanja pricesa u izvršavanju na drugu mašinu
- Skup proces, ali ako imamo razloge:
  - Bolje performanse (CPU duljina reda ili iskorištenje)
  - Bolja fleksibilnost

Migracija koda

- Ušteda u komunikaciji – dio koda prebaciti klijentu na izvršavanje
- Web crawler – mobilni agenti
- JIT instalacija klijenskog softwarea



Modeli migracije koda

- Samo kod
- Procesa u izvršavanju :
  - Model procesa od 3 segmenta:
  1. Segment koda
  2. Segment resursa
  3. Segment izvršavanja