Uvod u distribuirane informacijske sustave

Arhitekture distribuiranih sustava

Distribuirani sustavi

- Praktički svi današnji veliki sustavi su distribuirani sustavi
- Distribuirana obrada podataka
- Distribuirano programsko inženerstvo

Vrste informacijskih sustava

- Osobni sustavi
  - Nedistribuirani sustavi koji se izvršavaju na jednoj radnoj stanici
- Ugrađeni sustavi
  - Distribuirani sustavi koji se izvršavaju na jednom ili integriranoj grupi procesora
- Distribuirani sustavi
  - Pogone se na „labavo integriranoj” grupi procesora povezanih računalnom mrežom

Distribuirani sustavi - ciljevi

- Dijeljenje resursa
- Otvorenost
- Konkurentnost
- Skalabilnost
- Tolerancija grešaka

Distribuirani sustavi - zamke

- Kompleksnost
- Sigurnost
- Upravljivost
- Nepredvidivost

Arhitektura

- Arhitektura je fundementalna organizacija sustava utjelovljena s
  - Komponentama (građevnim blokovima sustava)
  - Vezama među blokovima i okolinom
  - Principima koji vode ka dizajnu i razvoju sustava
- Arhitekturalni stil je koherentni skup dizajnerskih odluka koji se tiču arhitekture
  - Tipične kombinacije kompozicije
  - Tipične odluke za veze komponente i ponašanja
  - ... generičko rješenje klase problema

Stilovi arhitekture

- Dizajniranje ili prihvaćanje arhitekture je neophodno za uspješan razvoj velikih sustava
- Stil – u smislu komponenti
  - Što su komponente
  - Kako se povezuju
  - Kako razmjenuju podatke
  - Kako se konfiguriraju u cijeli sustav
- Komponenta: modularna jedinica se sučeljem, zamjenjiva
- Konektor: mehanizam koji olakšava komunikaciju, koordinaciju i kooperaciju među komponentama

Stilovi arhitekture

- Slojevita (layered)
- Temeljena na objektima(object based)
- Usmjerena prema podacima (data centered)
- Temeljena na događajima (event drieven)

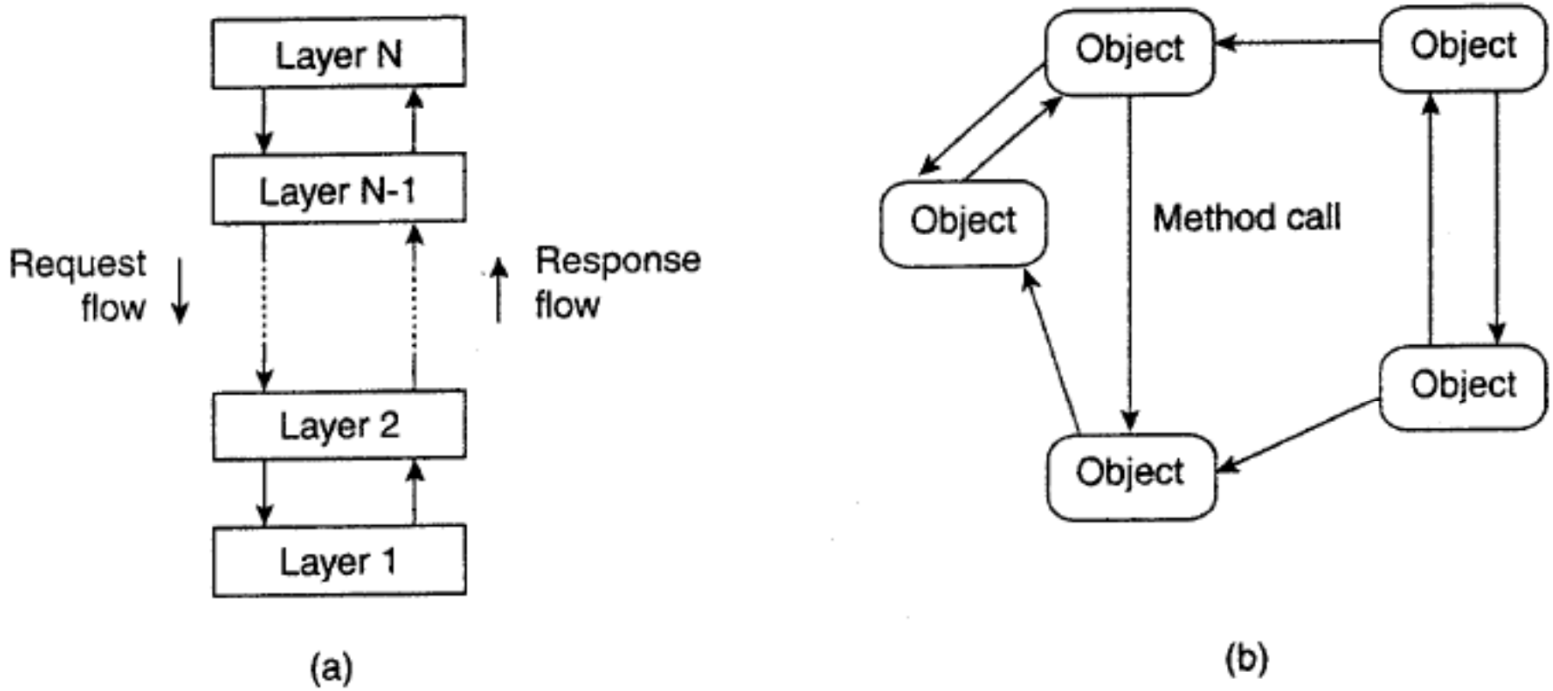
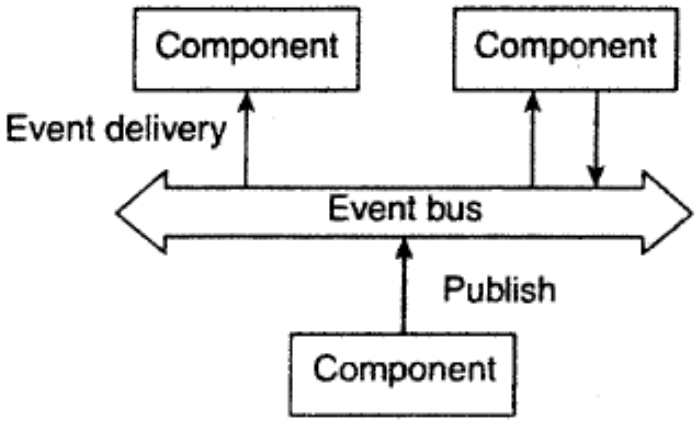
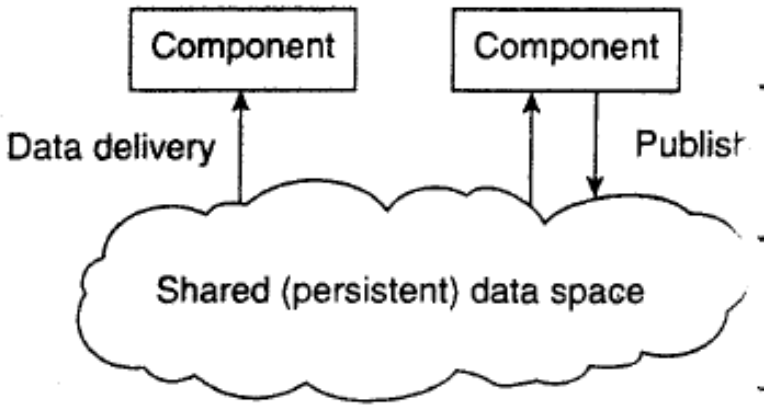


Figure 2-1. The (a) layered and (b) object-based architectural style.


Component
Component
Event delivery
Event bus
Publish
Component

(a)


Component
Component
Data delivery
Publish
Shared (persistent) data space

(b)

Arhitekture sustava

- Klijent server
  - Klijenti traže usluge od servera.
  - Serveri pružaju usluge.
  - Razdvojenost uloga
- Distribuirani objekti
  - Nema radvajanja klijenata i severa
- Pear to pear
- Hibridne

Middleware

- ✓*The slash in client/server*
- ✓*The dash in client-server*
- ✓*The glue between components*

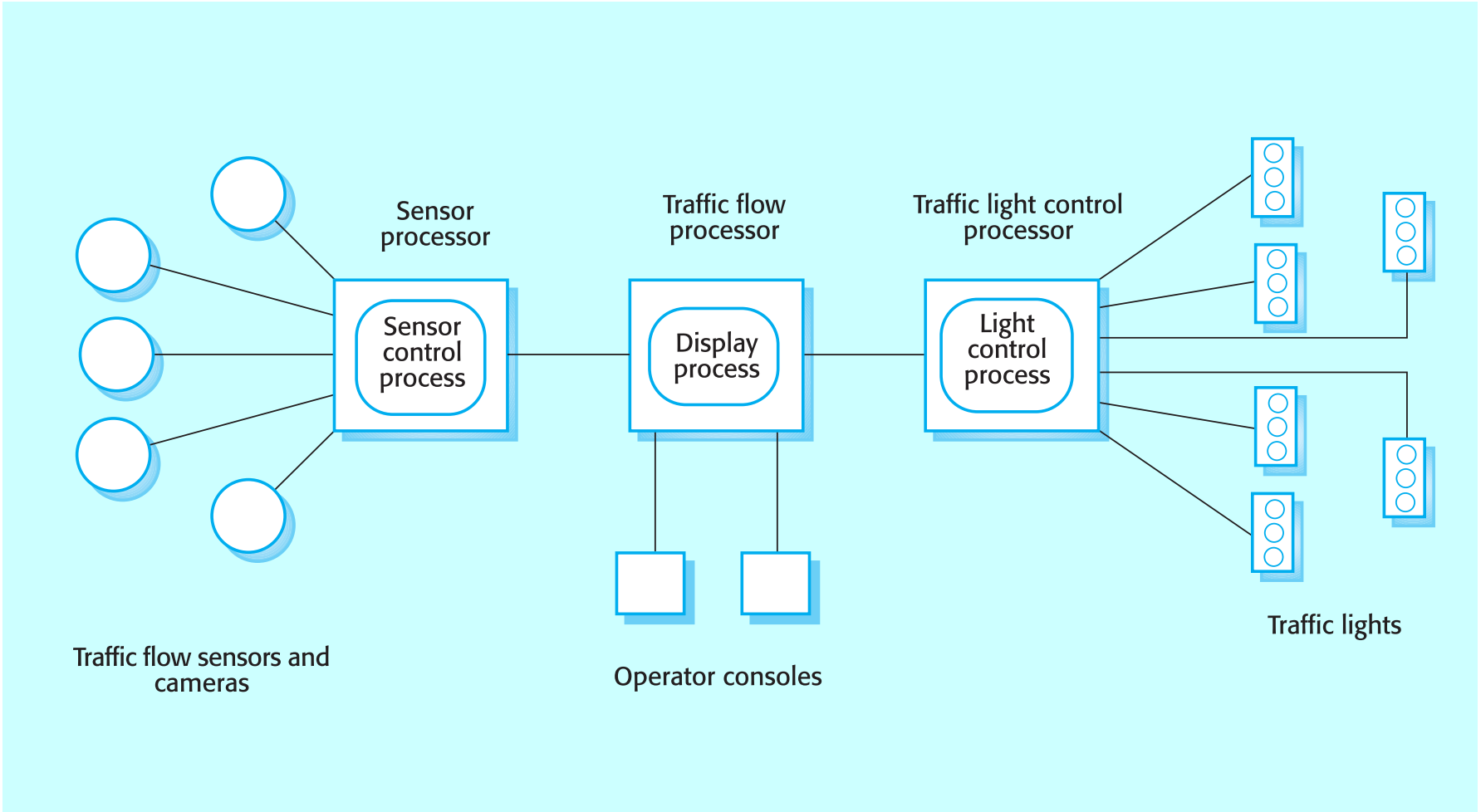
- Dio programa koji podržava različite komponente distribuiranog sustava
- Centar sustava
- Često gotova komponenta
  - TPM (transaction processing monitor)
  - Data converters
  - Communication controllers

Arhitekture više procesora

- Najjednostavniji model distribuiranog sustava
- Nekoliko procesa koje se mogu, ali ne moraju izvršavati na različitim procesorima

- Distribucija procesa se može predefinirati ili se može raspoređivati za vrijeme izvršavanje programa (dispatcher)

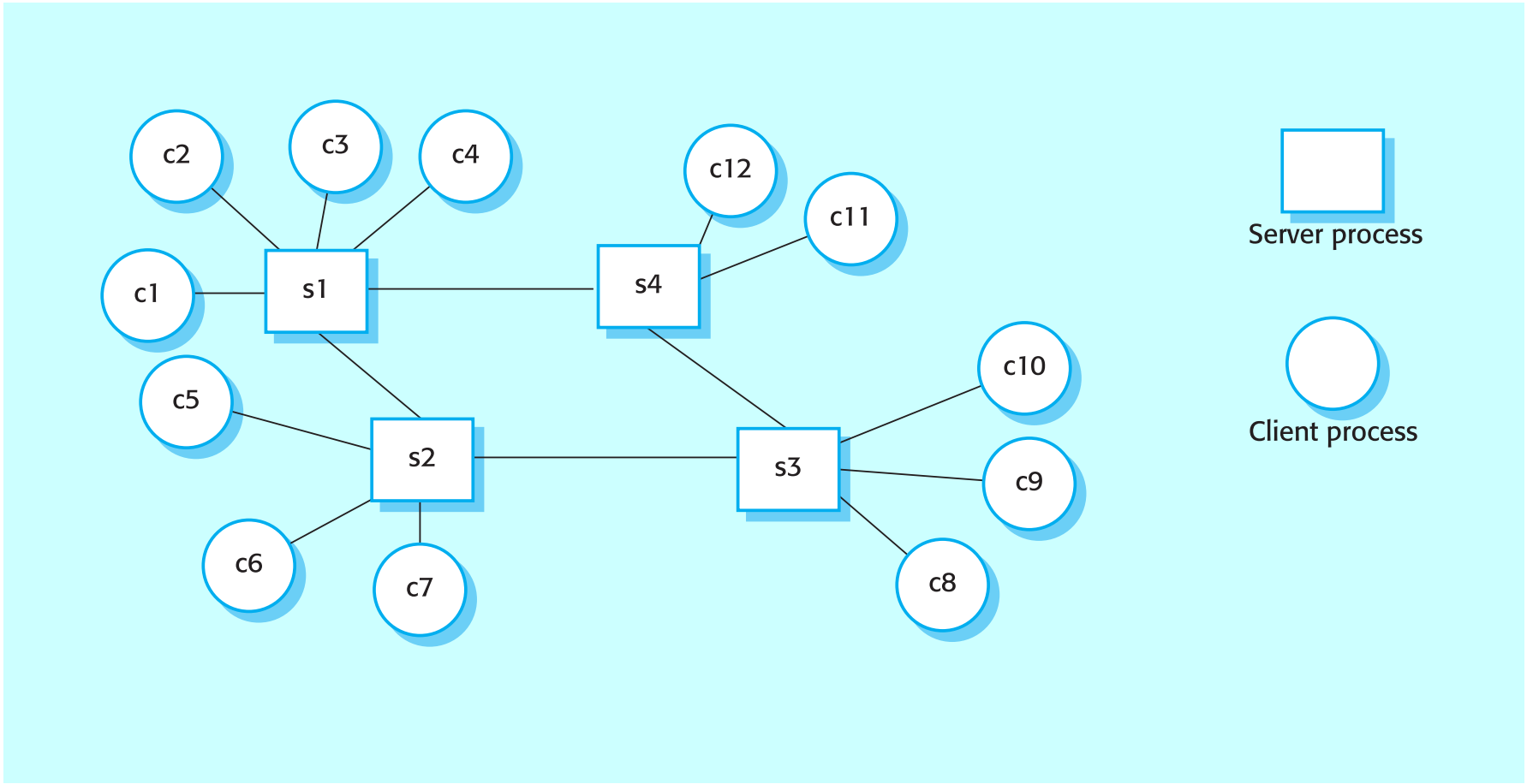
Primjer : upravljanje semaforom



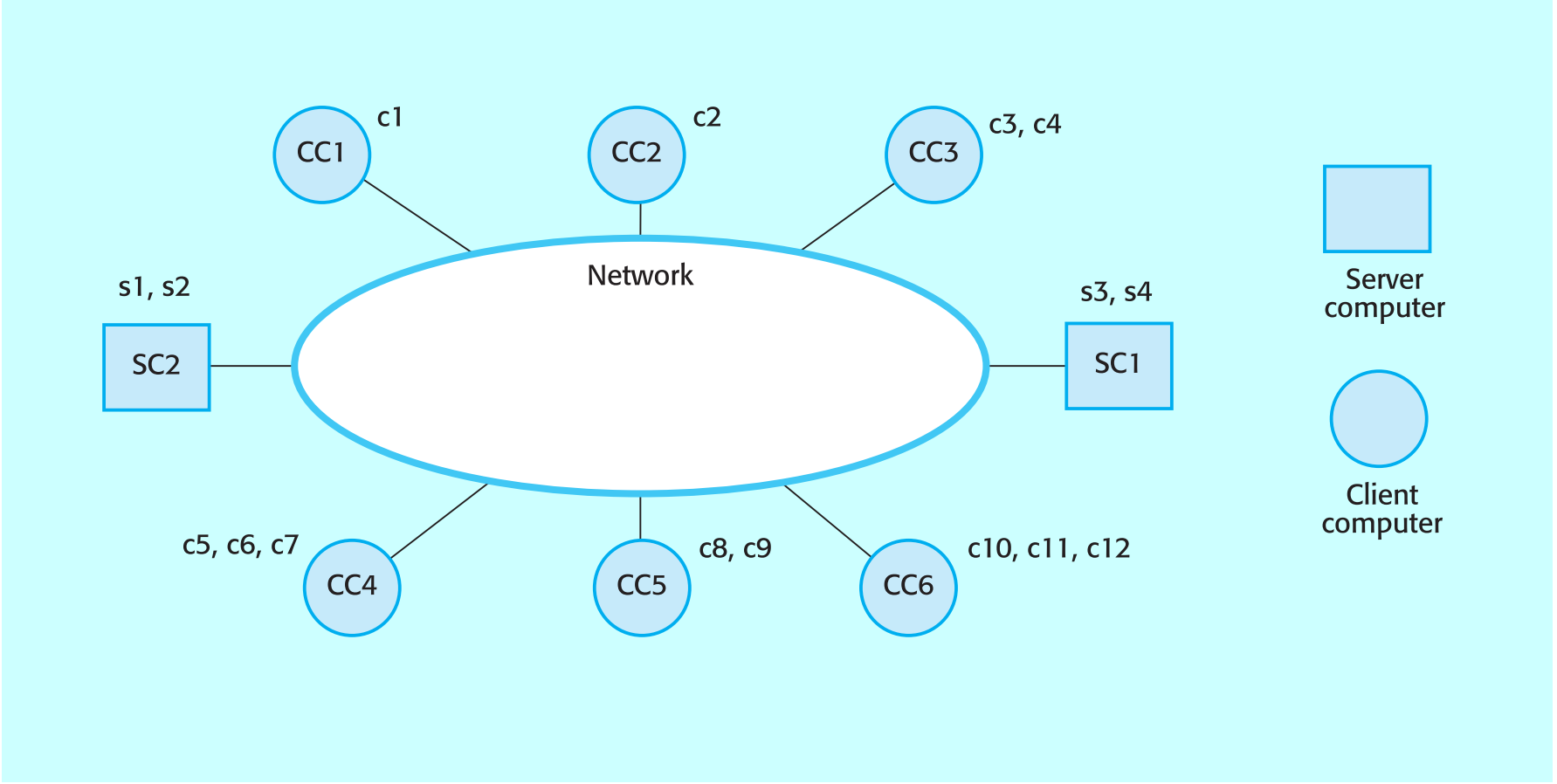
Klijent –server arhitekture

- Aplikacija = skup usluga koje nude serveri i skup klijenata koji te usluge koriste
- Klijent znaju za servere, ali serveri ne moraju znati za klijente
- Klijenti i serveri su logički procesi
- Ne mora biti 1:1

Klijen server sustav

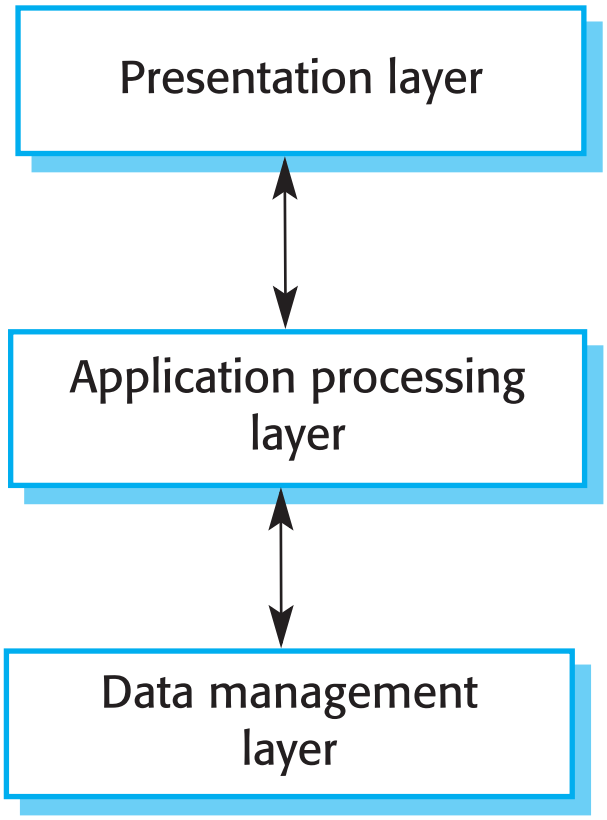


Računala u klijet-server mreži



Slojevita arhitektura aplikacije

- Prezentacijski sloj:
  - Komunikacija sa korisnicima
- Aplikacijska obrada:
  - Funkcionalnost specifične aplikacije
- Upravljanje podacima:
  - Upravljanje bazom podataka

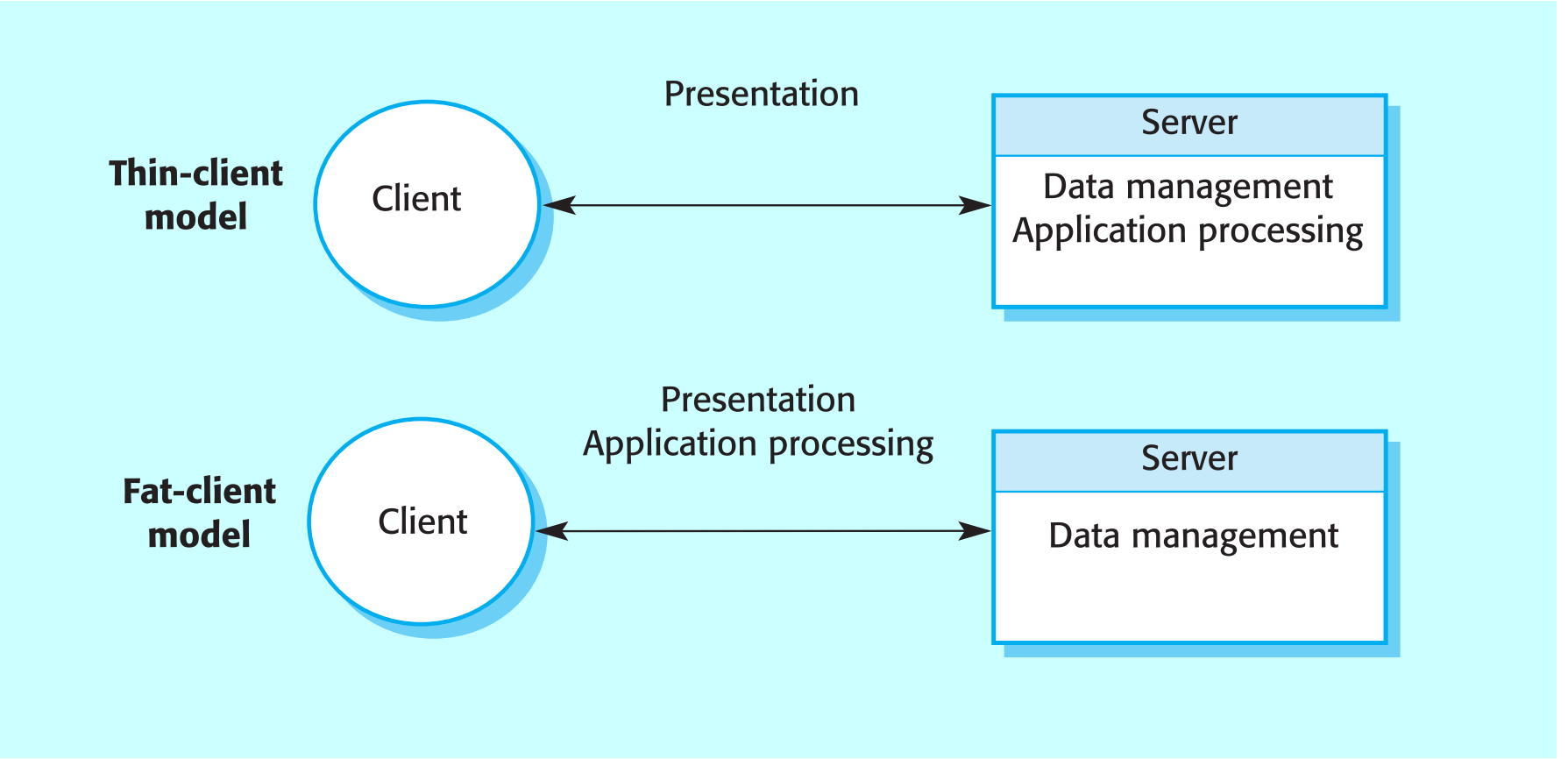


„mršavi” i „debeli” klijenti

- Model Thin-client:
  - Obrada podataka uglavnom sa strane servera
  - Služi samo za prezentaciju
- Model Fat-client
  - Dio obrade podataka odrađuje klijent
  - Server se brine samo o bazi podataka

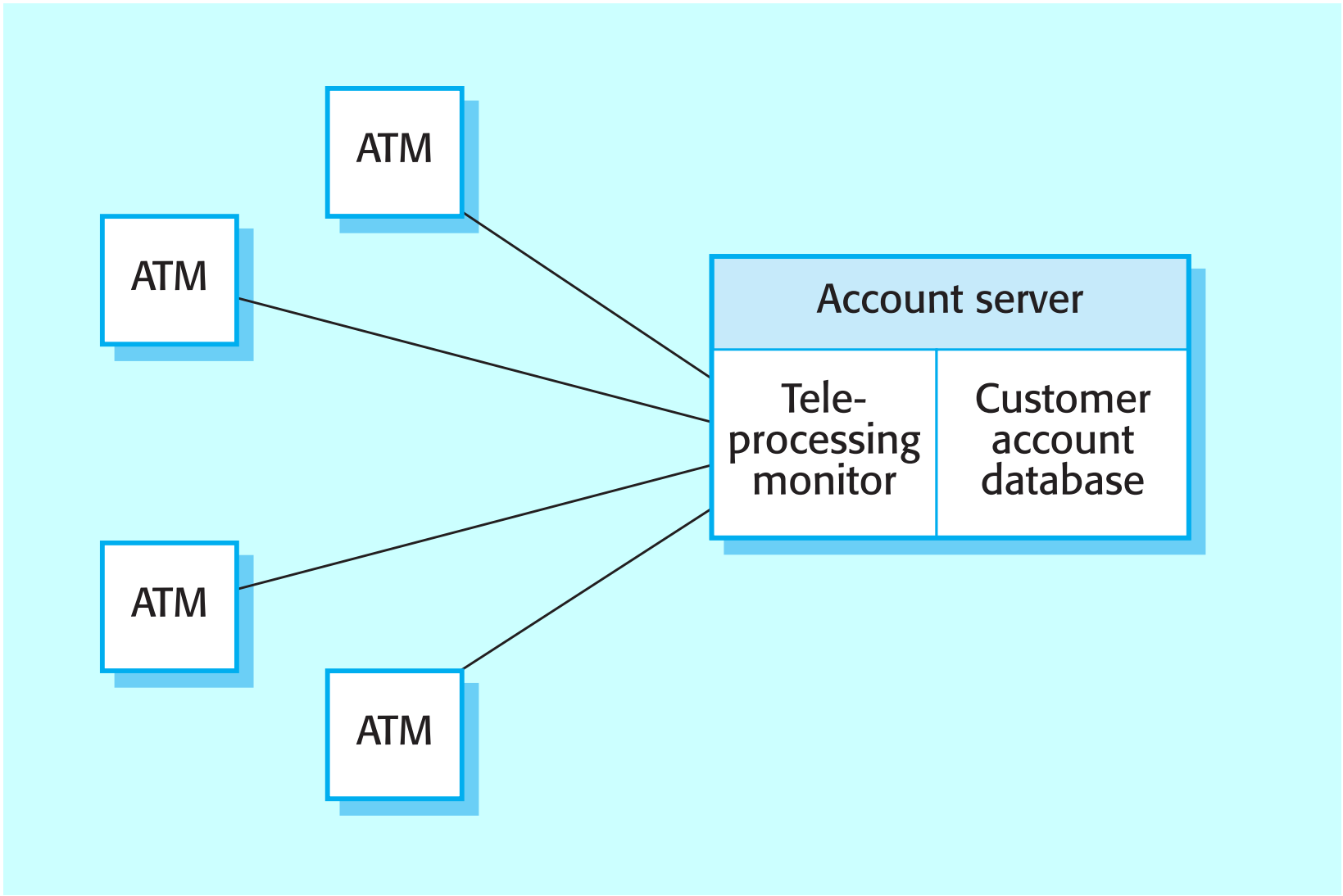
„two-tier” arhitekture

„mršavi” i „debeli” klijenti



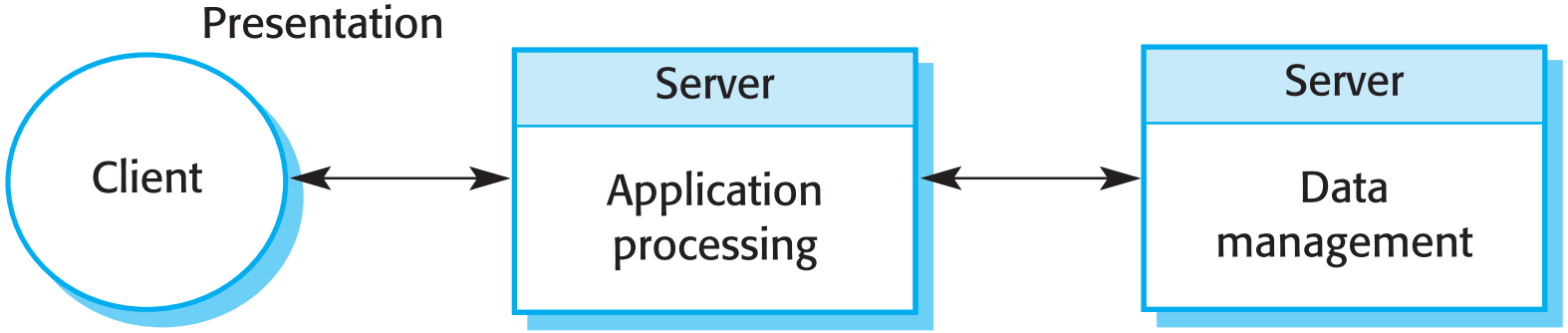
„mršavi” i „debeli” klijenti

- Mršavi klijenti
  - Koriste se kod prilagodbe „običnih starih ” sustava u distribuirane - aplikacija se ponaša kao server, a sučelje sa korisnikom kao klijent
  - Mana: veliki zahtjevi za server i mrežu
- Debeli klijenti
  - Dio obrade se izvršava na klijentu
  - Pogodno za “nove” klijent-server sustave kada su sposobnosti klijenta unaprijed poznate
  - Kompleksniji sustav, klijenti moraju imati instalirane novije verzije aplikacije

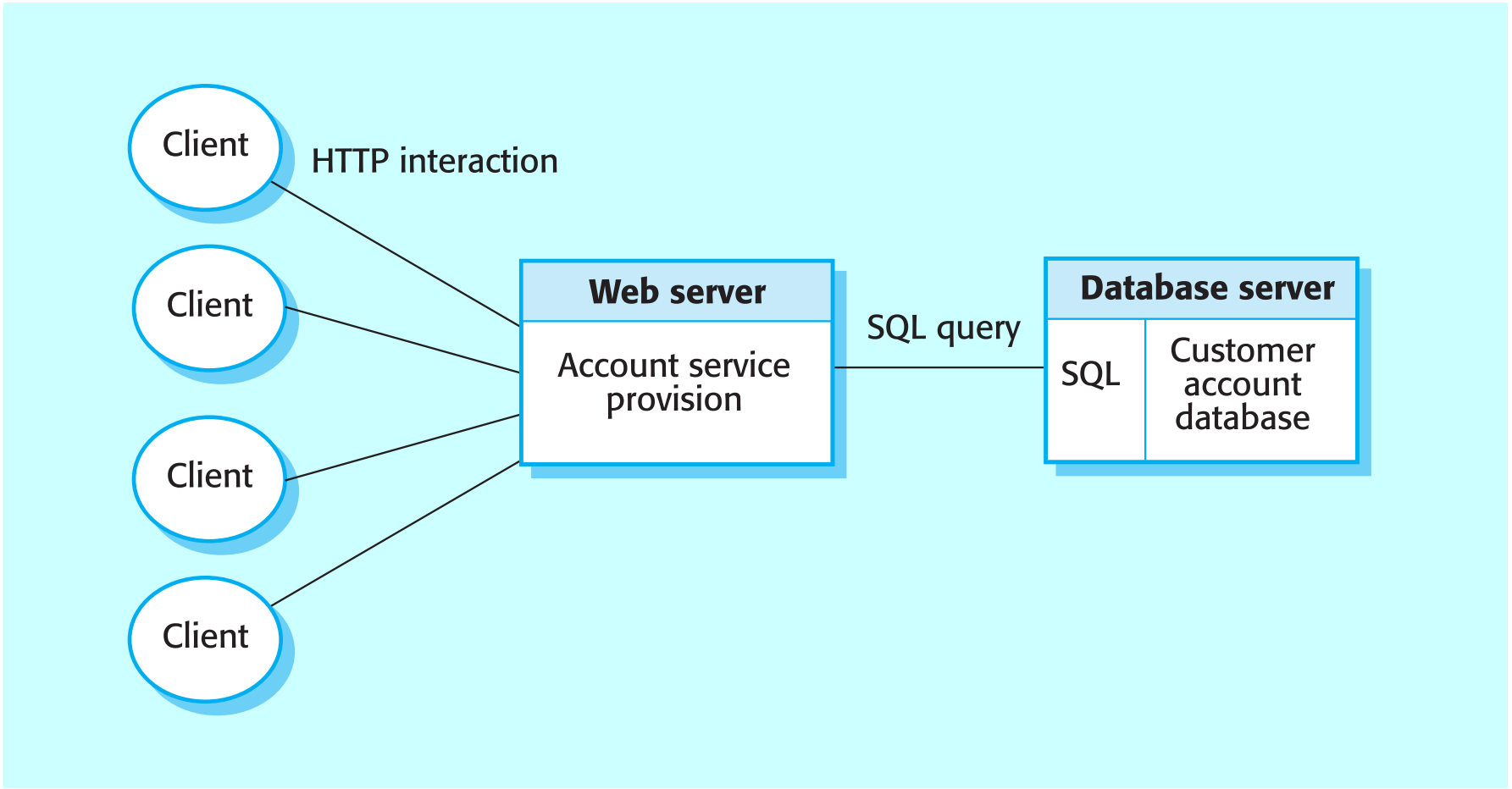

ATM
ATM
ATM
ATM
Account server
Tele-processing monitor
Customer account database

3-tier arhitektura

- 3 sloja – 3 procesora?
- Svaki sloj arhitekture aplikacije izvršava se na drugom procesoru
- Izbjegavaju se problemi 2-tier arhitektura (opterećenje servera i mreže, i kompleksnost upravljanja)
- Skalabilnost – kako zahtjevi rastu, dodaju se novi serveri



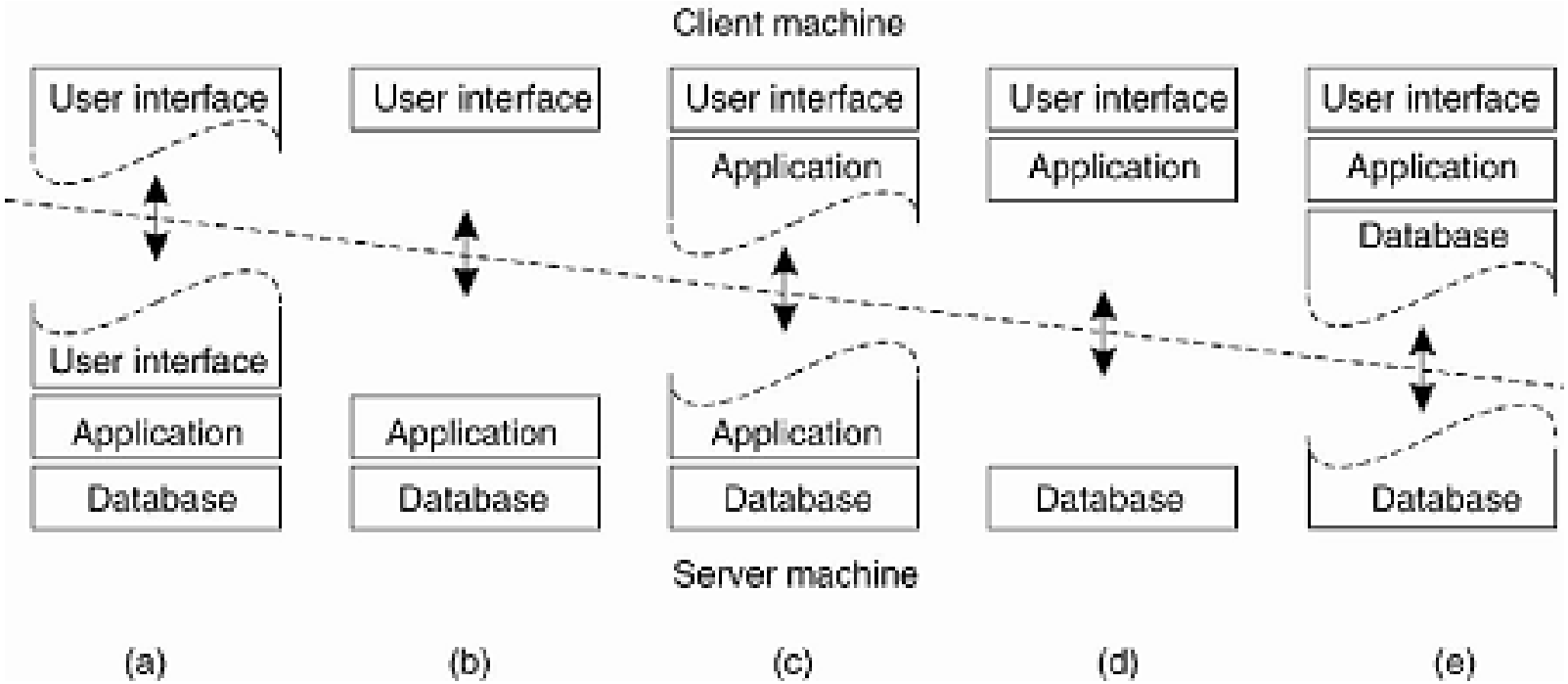
Primjer: internet bankarstvo



Korištenje klijent server arhitektura

arhitektura	Primjena
2-tier, „thin client”	Starije, nedistribuirane kod kojih su obrada podataka i aplikacija nerazdovjive Računski zahtjevne aplikacije (npr. kompajleri) sa malo obrade podataka Apliakcije koje rade sa velikim količinama podataka (pretraživanje, upiti) sa malo aplikacijske obrade
2-tier, „fat client”	Aplikacije kod kojih se obrada izvršava sa nekim gotovim programom Gdje je potrebno računski zahtjevna obrada podataka (vizualizacija, renderiranje) Apliakcije sa relativno stabilnom funkcionalnošću krajnjeg korisnika
3-tier ili multi-tier	Velike aplikacije sa velikom brojem klijenata Aplikacije gdje su podaci i aplikacije nestabilni Gdje se integriraju podaci iz više izvora

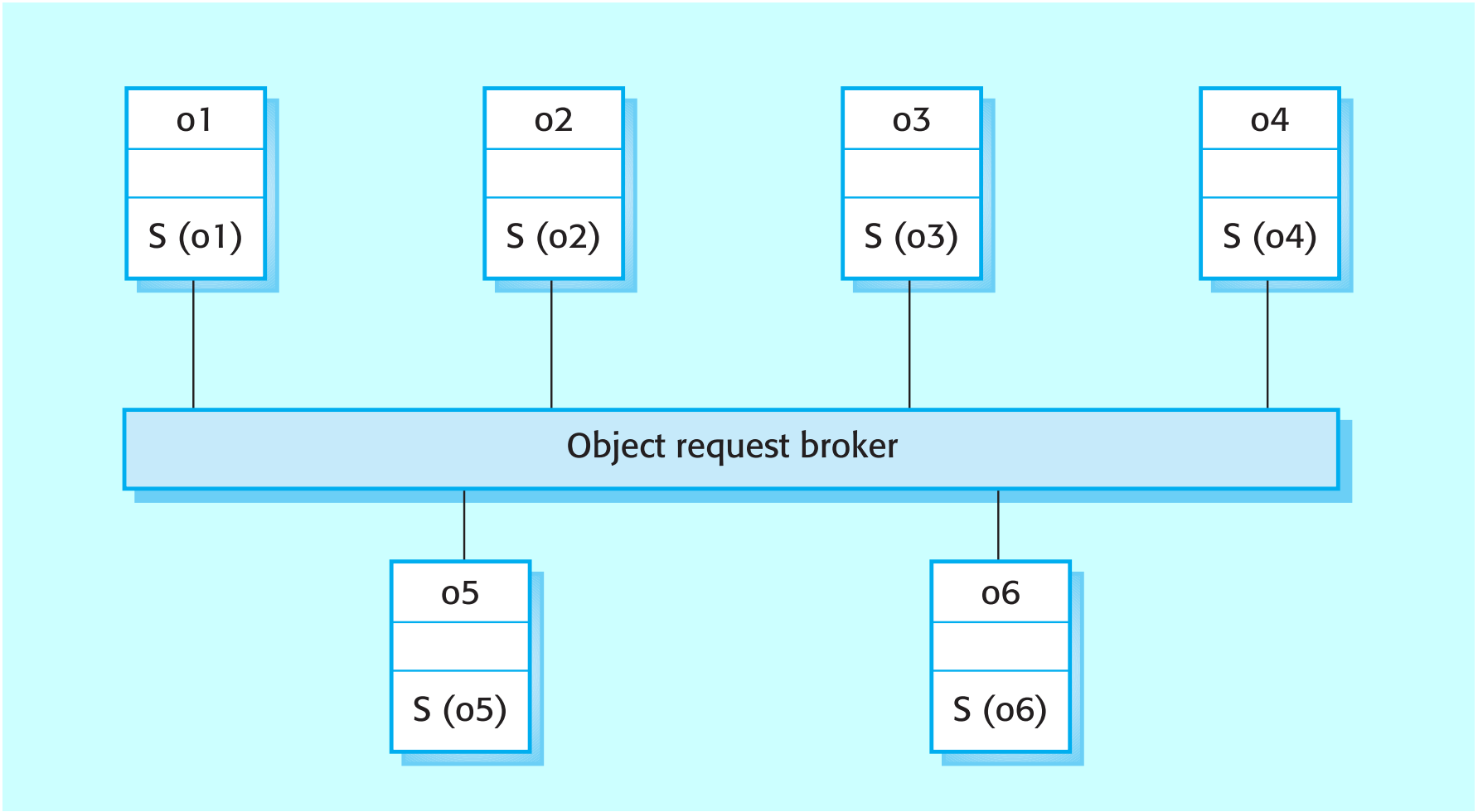
N-tier arhitektura



Arhitektura distribuiranih objekata

- Nema razlike između klijenata i servera
- Svaki entitet je objekt koji pruža usluge drugim objektima i prima usluge od drugih objekata
- Komunikacja među objektima se odvija kroz middleware sustav - Object Request Broker
- Složeniji dizajn

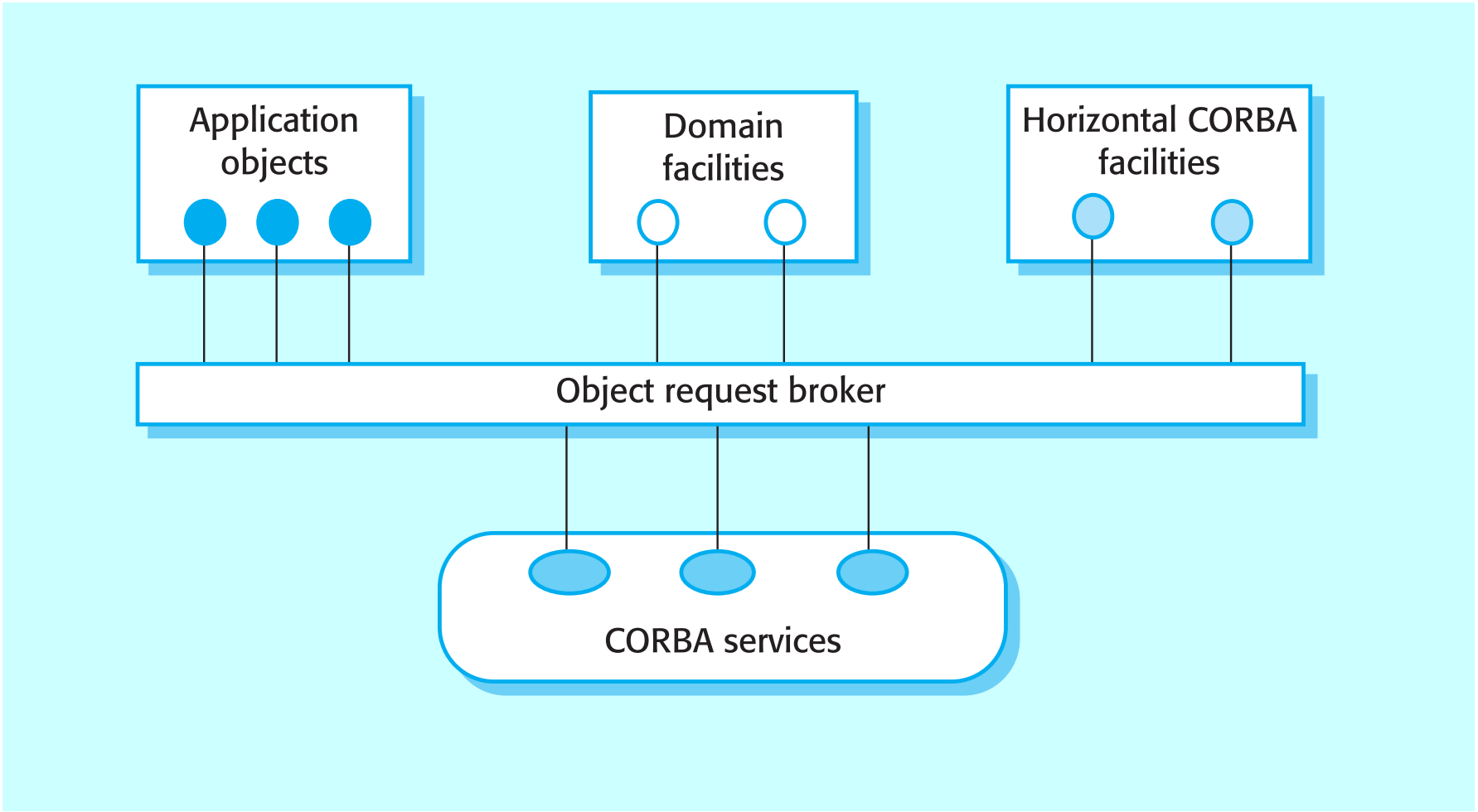
Distribuirani objekti



CORBA

- OMG – *Object Management Group, 1989*
  - Promoviranje objektne tehnologije
  - Usmjeravanje razvoja definiranjem referentnog modela
- CORBA - *Common Object Request Broker Architecture*
  - Univerzalni middleware koji
    - Omogućava kreiranje objekata koji s drugim objektima komuniciraju bez da znaju gdje se nalaze
    - OO sw komponente u heterogenim distribuiranim okruženjima

Struktura CORBA aplikacije



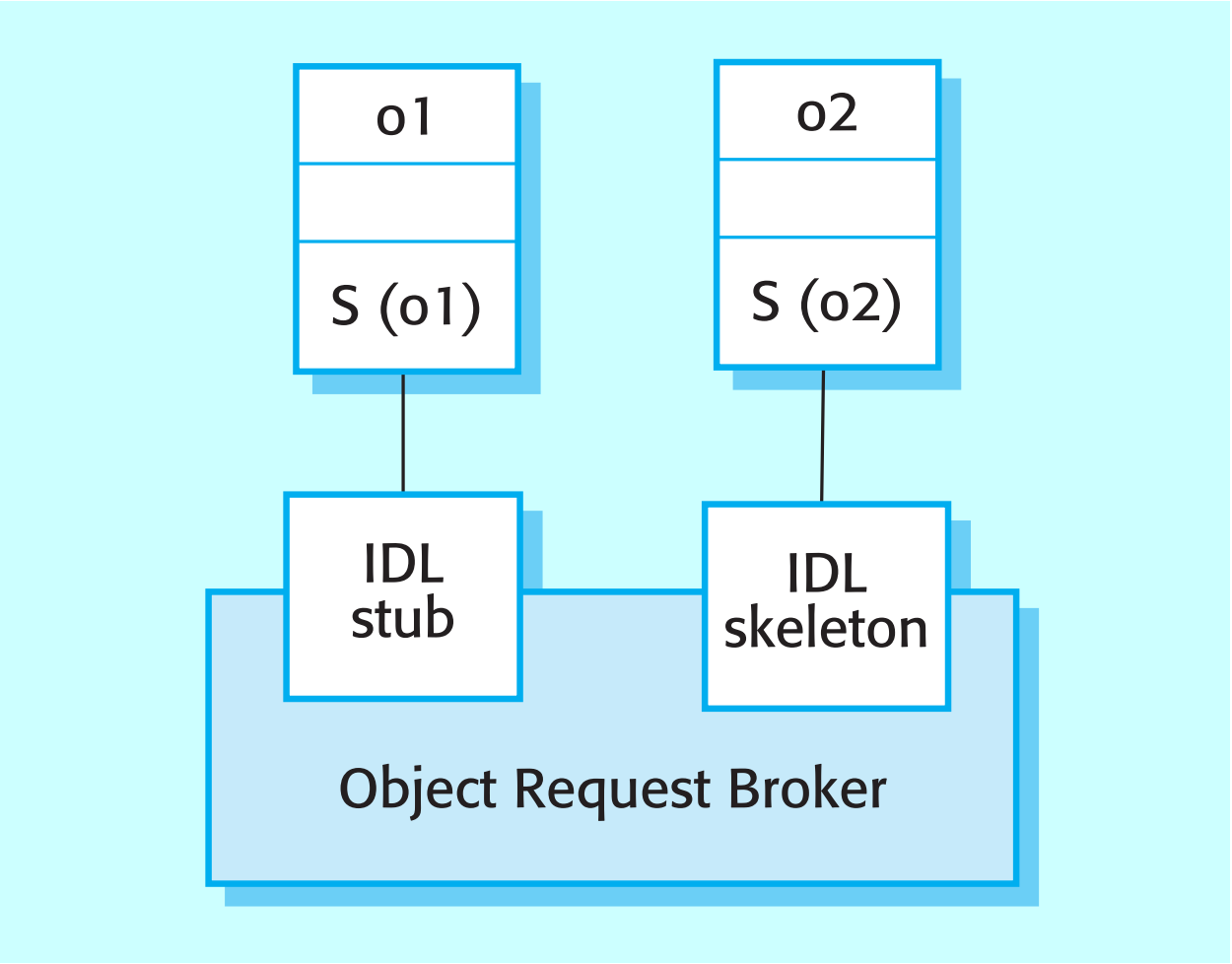
CORBA definira

- Objektni model za aplikacijske objekte
  - CORBA objekt je enkapsulacija stanja dobro definiranog sučelja definiranog u IDL
- ORB – broker koji upravlja zahtjevima objekata
- Skup generalnih servisa
  - Imenovanje, „trgovina”, notifikacije, transakcije
- Skup zajedničkih komponenti koje su nadograđene na te servise

CORBA objekti

- CORBA objekti
  - Slični objektima u C++ ili Javi
  - Moraju imati izdvojeno sučelje definirano u idl jeziku koji je sličan c++
  - Postoji mapiranje IDL-> druge programske jezike (C++ili Java) tako da objekti realizirani u različitim jezicim amogu komunicirati
- ORB
  - Upravlja komunikacijom među objektima

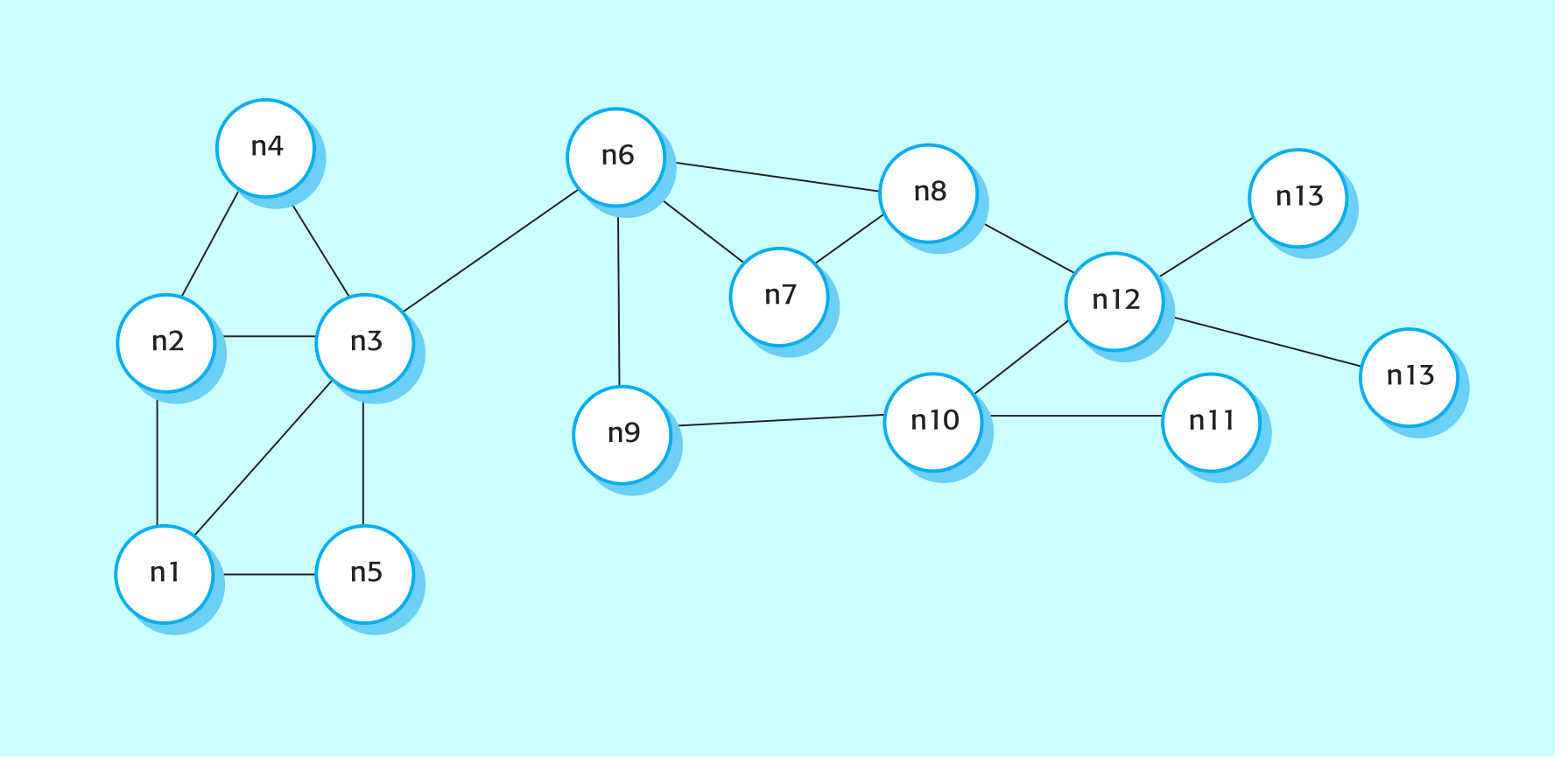
Komunikacija pomoću ORB



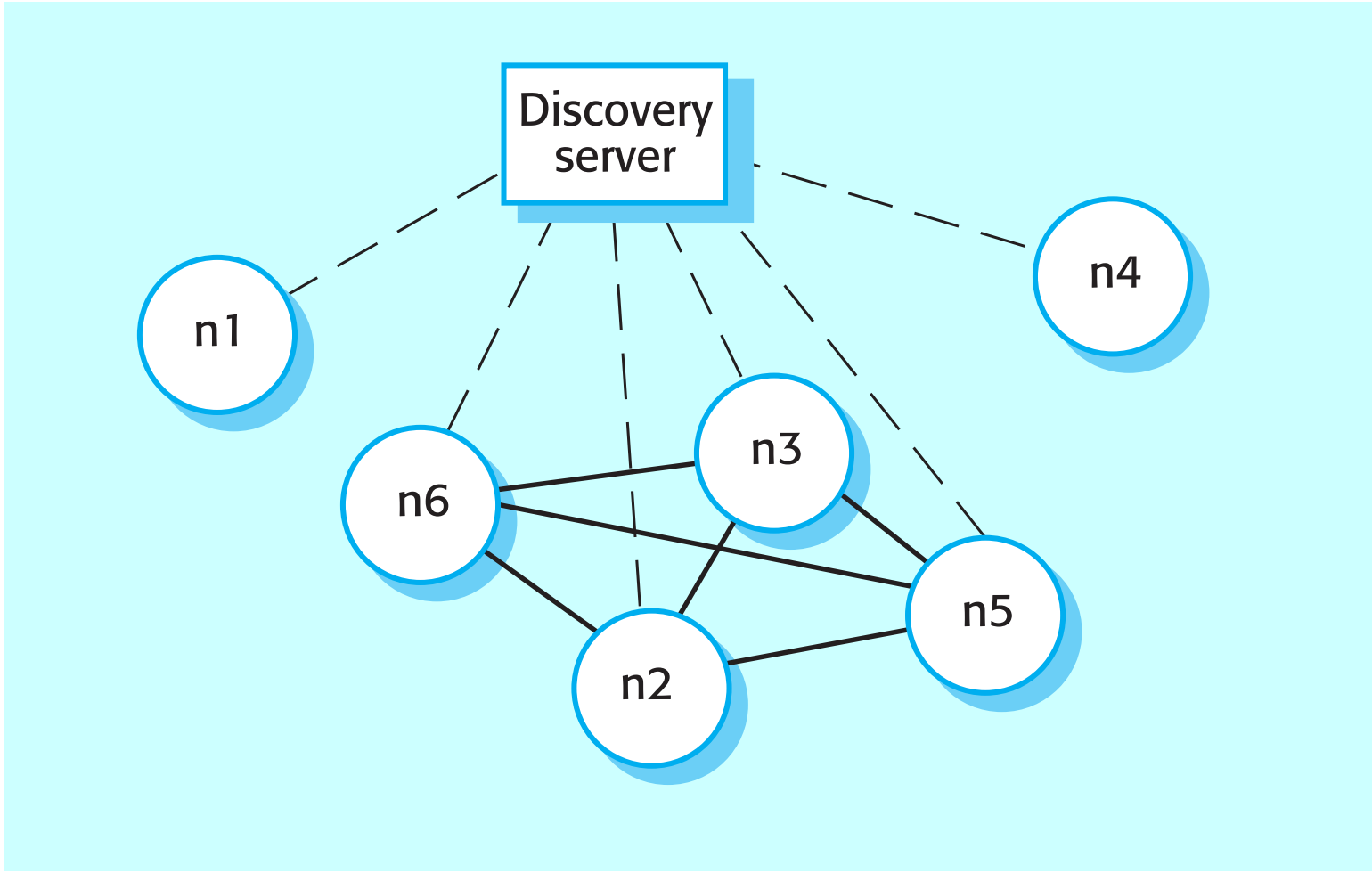
Pear to pear

- Decentralizirani sustavi
- Kooperacija
- Simetrija uloga – svaki čvor je istovremeno klijent i server
- Sustav se dizajnira na način da se iskoristi prednost računalne snage i pohrane velikog broja umreženih računala
- Prvi p2p - Napster
- Logičke mrežne arhitekture
  - Decentralizirane
  - Polucentralizirane
- Arhitekture aplikacije
  - Generičke organizacije komponeniti koje čine p2p

Decentralizirane p2p arhitekture



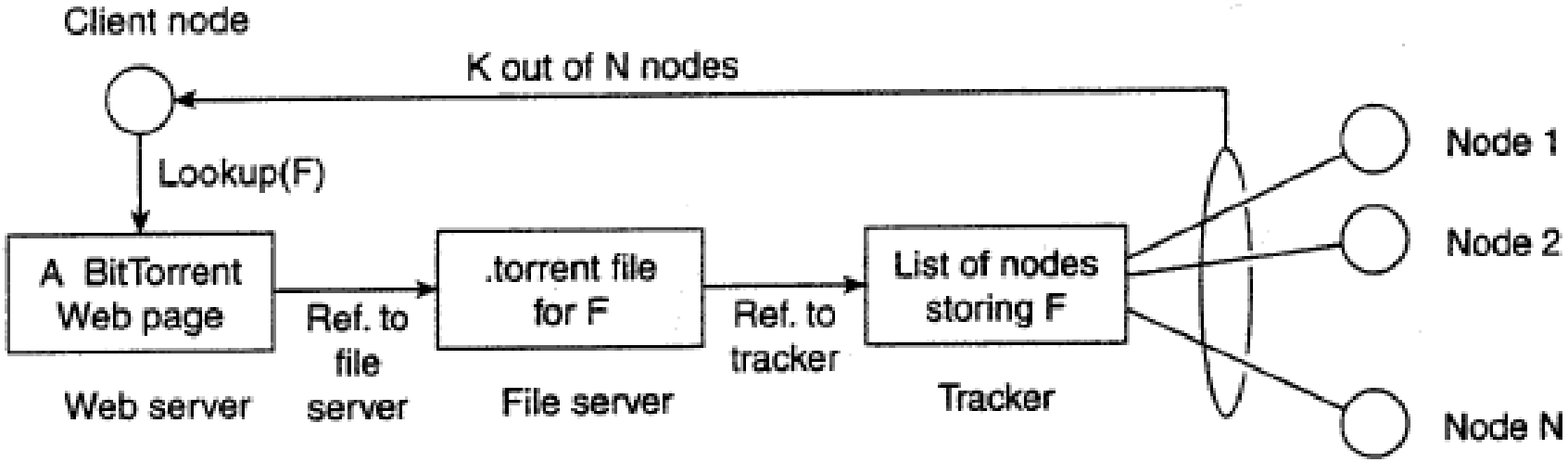
Polu-centralizirane p2p arhitekture



Hibridna arhitektura
primjer: .bitTorrent

- .bitTorrent - primjer kolaborativnog distribuiranog sustava
- Razmjena datoteka
- Pear to pear downloadanje podataka
- Krajnji korisnik dohvaća dijelove datoteke od nekoliko različitih korisnika
- Dijelovi datoteka se spajaju u konačni potpuni dokument

.bitTorrent



- Web stranica : zna lokaciju .torrent datoteke
- .torrent datoteka: referenca tracker
- Tracker: zna listu čvorova koji imaju traženu datoteku
- Čvorovi: dozvoljavaju klijentu download dijelova datoteke

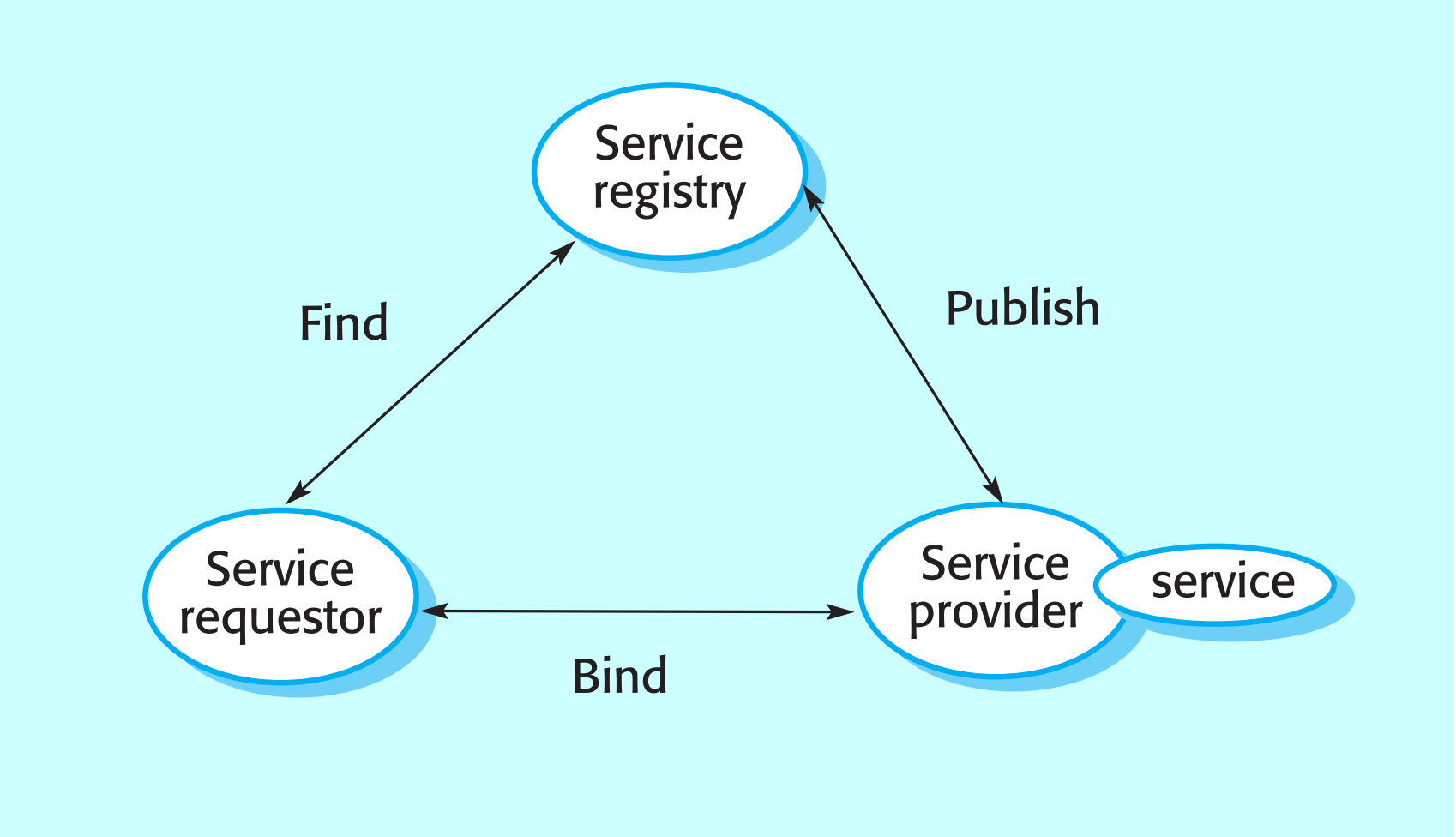
.bitTorrent

- Pravila koja „prisiljavaju” na suradnju
- Korisnik može downloadati samo ako omogući upload
- Npr. Q downloada podatke od P
- Kada čvor P primjeti da Q downloada više nego uploada, smanjuje brzinu kojom šalje podatke prema Q

SOA – *Service oriented architectures*

- SOA se temelji na ideji korištenja usluga „iz vana”
- Web servis je standardni pristup kreiranju reiskoristive komponente kojoj se može pristupiti i koristiti je putem web-a
- *Service definition : An act or performance offered by one party to another. Although the process may be tied to a physical product, the performance is essentially intangible and does not normally result in ownership of any of the factors of production.*

Web servis


Service registry
Find
Publish
Service requestor
Service provider
service
Bind

Standardi za web servise

- SOAP (*Simple object access protocol*) – protokol definira strukturiranu razmjenu podataka među web servisima
- WSDL (*Web service description language*) - kako se predstavljaju sučelja web servisa
- UDDI (*Universal description discovery and integration)* - kako se koriste informacije iz opisa servisa za pronalazak potrebnih servisa

Cloud computing

- Arhitektura za izradu visoko skalabilnih aplikacija
- Clusters
- Public cloud: nudi računanje, pohranu i komunikaciju i ... Naplaćuje
- Iluzija beskonačne skalabilnosti
- Amazon Web Services, Google App Engine, Microsoft Azure

Sljedeći put

- Procesi i niti