2. előadás

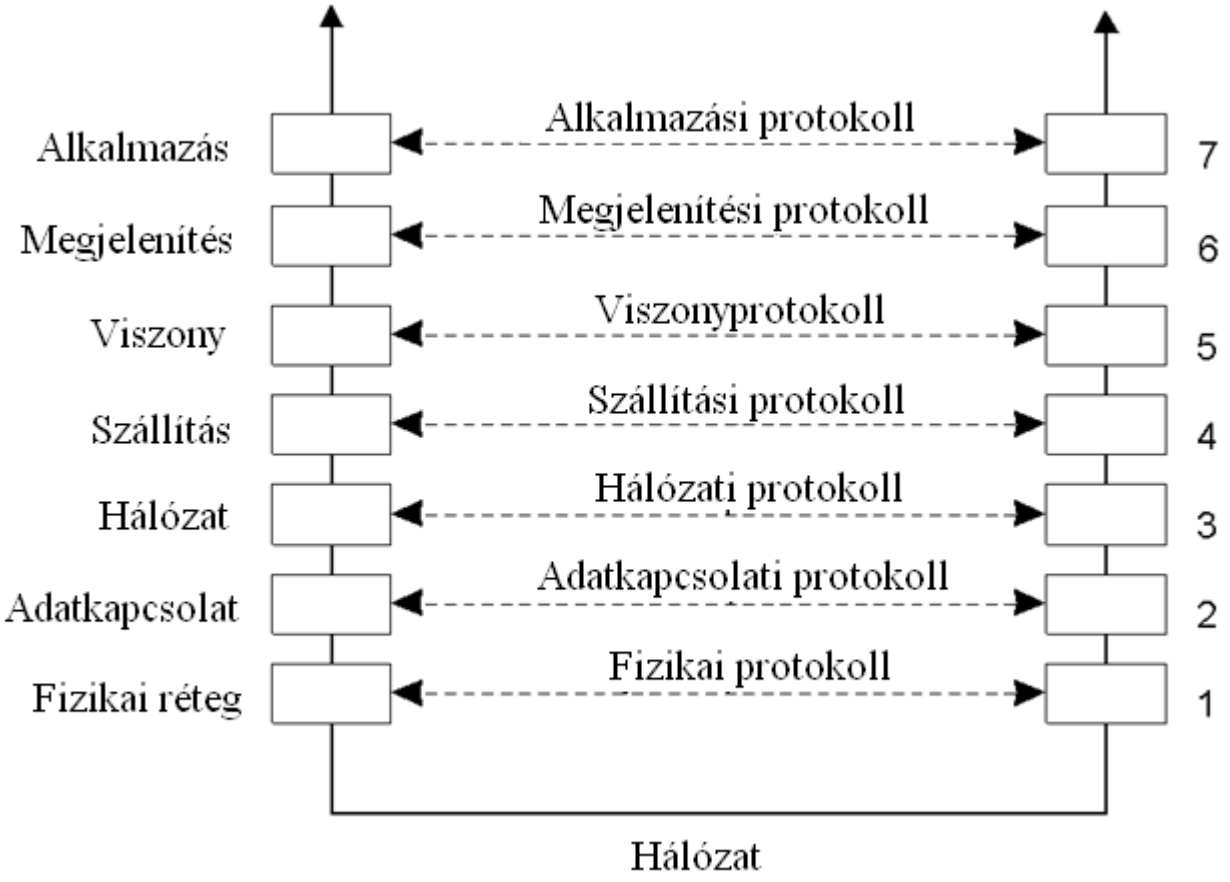
Kommunikáció

1. rész

Kommunikáció

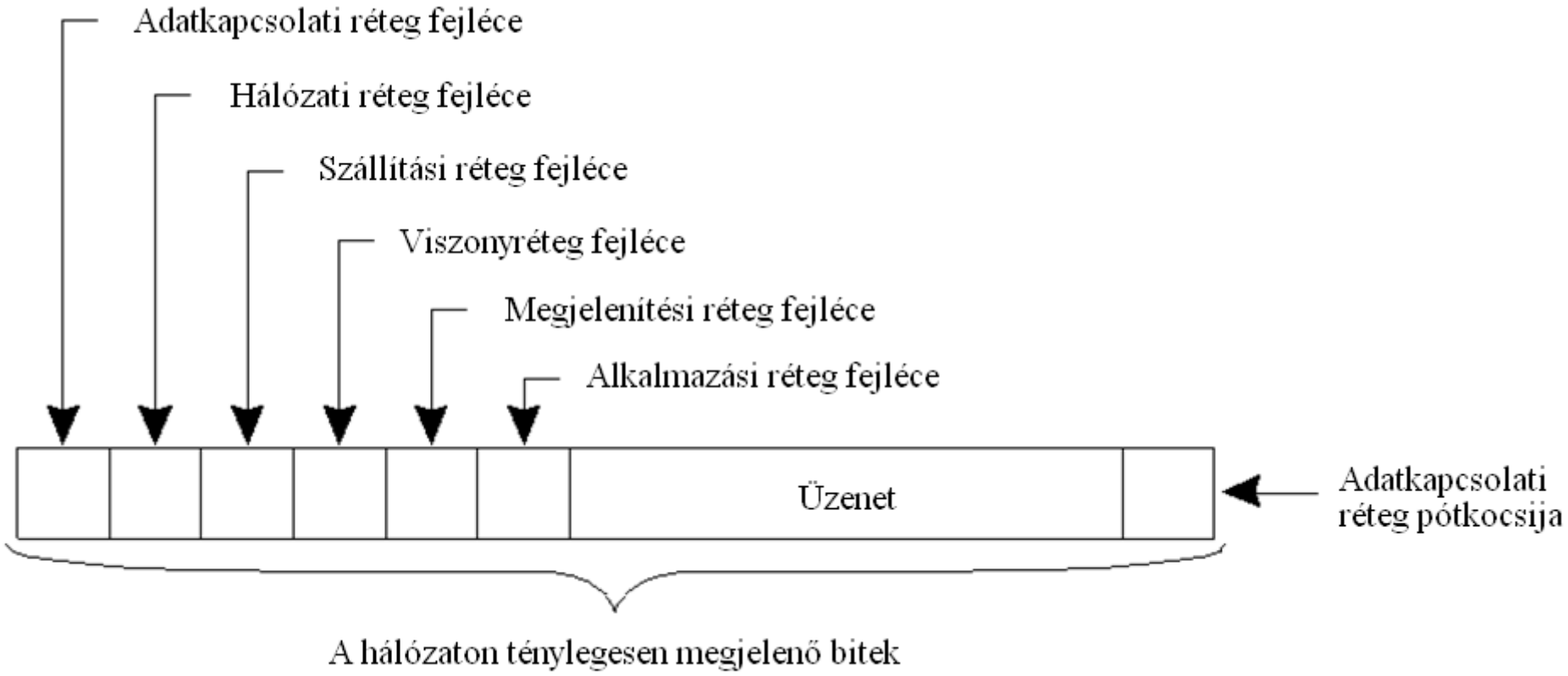
- „A” és „B” folyamatnak meg kell egyeznie a bitek jelentésében
- Szabályok – protokollok
  - ISO OSI
- Többrétegű protokollok előnyei
- Kapcsolat-orientált / kapcsolat nélküli

Protokollrétegek (1)



Az OSI-modell szintjei, interfészei és protokolljai.

Protokollrétegek (2)



A hálózaton megjelenő tipikus üzenet szerkezete.

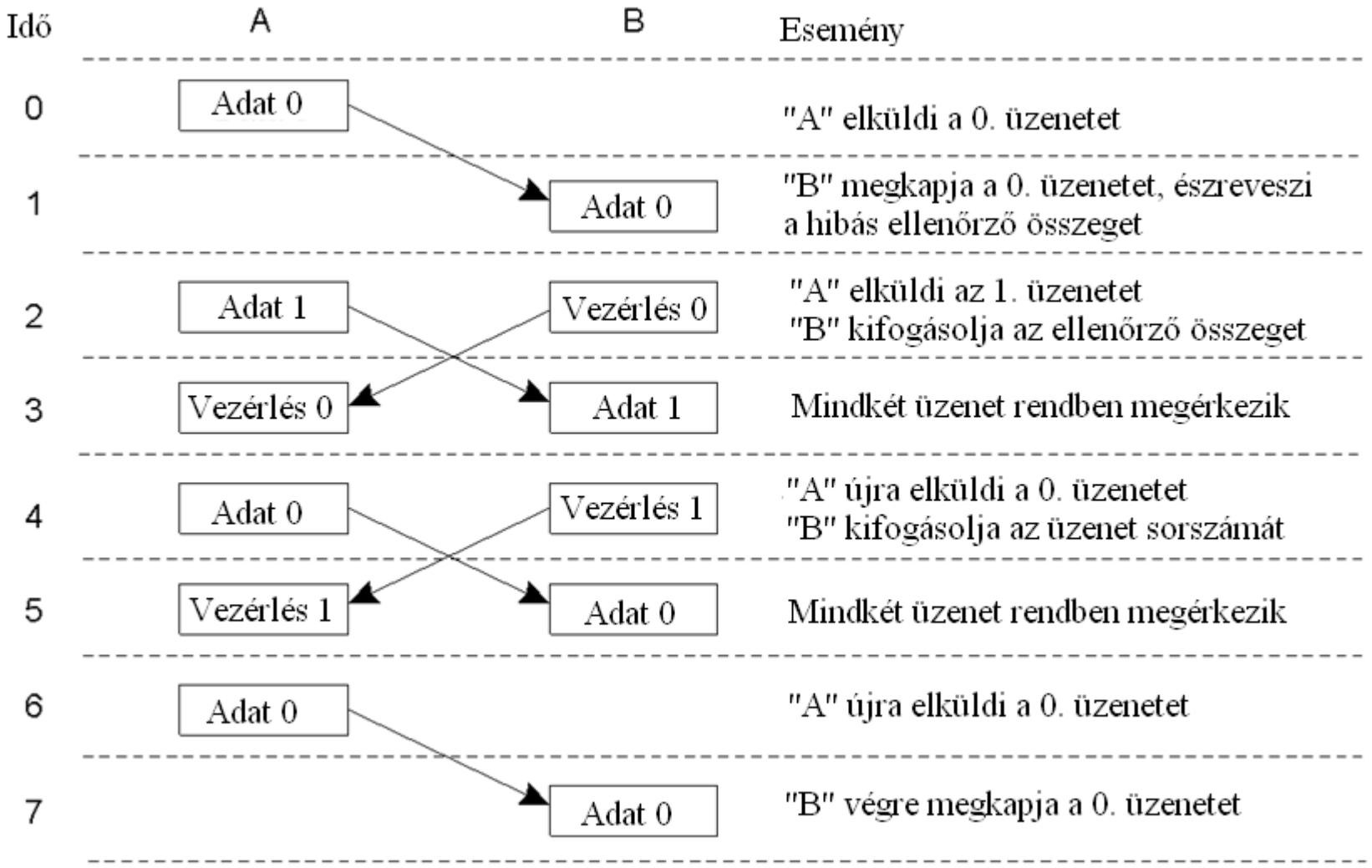
Fizikai réteg

- 0/1 átvitelének módjai
- Feszültség, küldés sebessége, stb.

Adatkapcsolati réteg

- Meghibásodást fedi el
- Adatkeretek
- Ellenőrző összeg

Adatkapcsolati réteg (2)



A küldő és a fogadó közötti párbeszéd az adatátviteli rétegben.

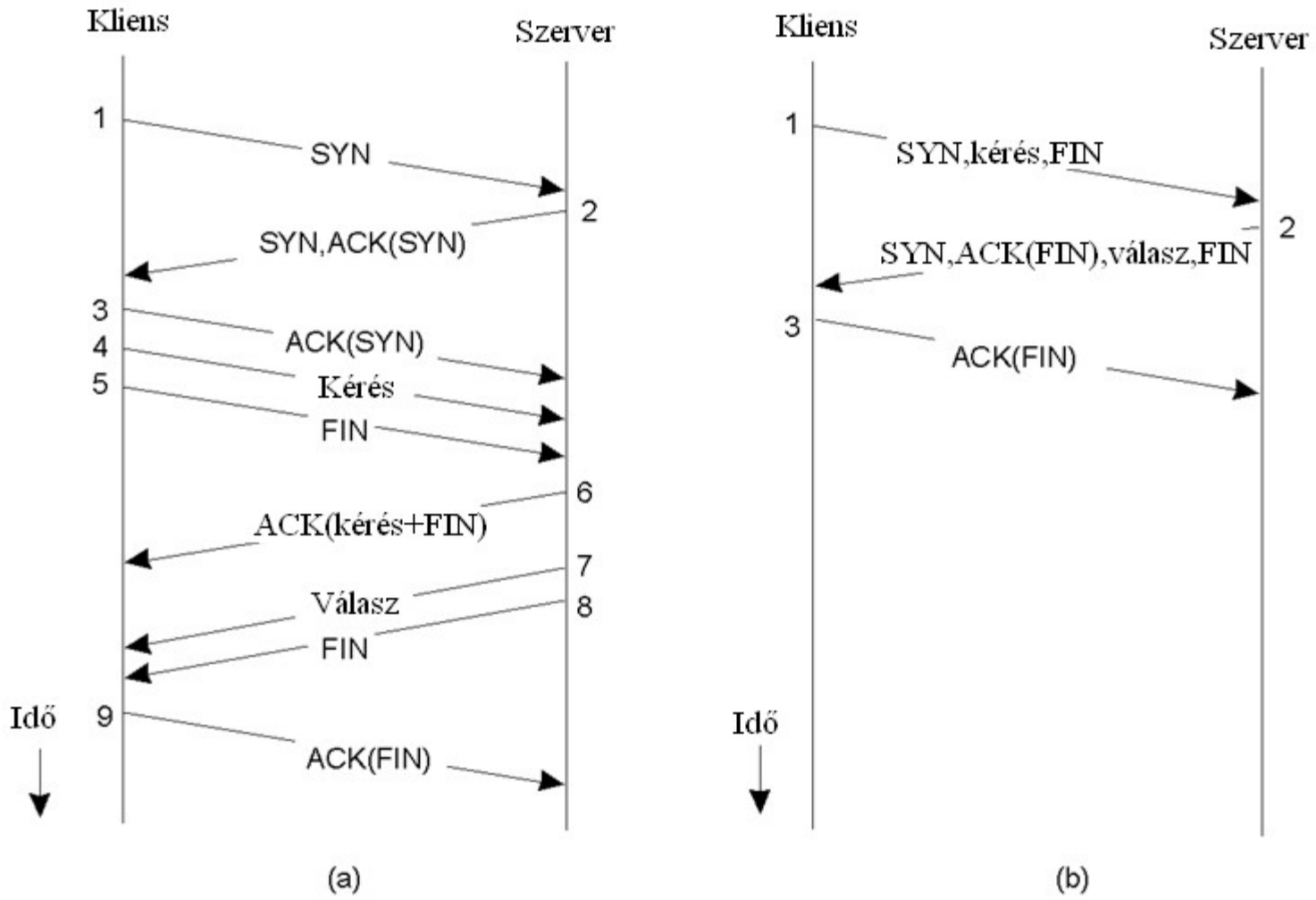
Hálózati réteg

- Útvonalválasztás
- IP – Internet Protocol

Szállítás

- Csomagok elveszhetnek
- Megbízható kapcsolatot épít
- TCP – Transmission Control Protocol
- UDP – Universal Datagram Protocol
- RTP – Real-time Transport Protocol

Kliens-szerver TCP



a) A TCP normál működése

b) Tranzakciós TCP

Viszony- és megjelenítési rétegek

- Párbeszéd vezérlése
- Bitek jelentése

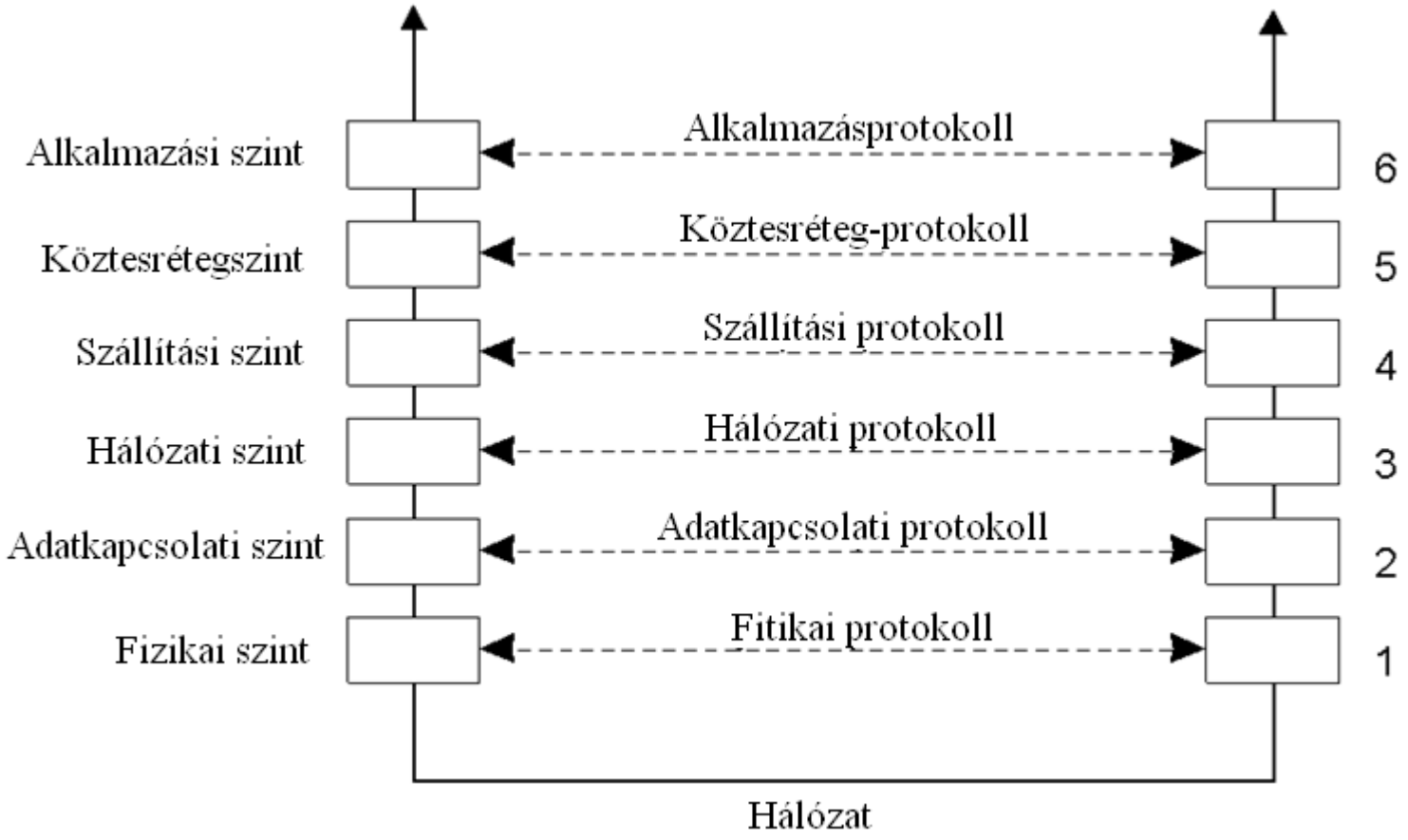
Alkalmazási réteg

- FTP – File Transfer Protokoll vs. ftp-program
- Köztesréteg-protokoll

Köztesréteg protokoll (1)

- Logikailag az alkalmazási rétegbe tartozik, de tartalmaz általános célú protokollokat
- Köztesréteg szolgáltatásait támogató protokollok
  - Azonosító
  - Jogosultságok
  - Véglegesítő protokoll
  - Elosztott zárolási protokoll
- Magas szintű kommunikációs szolgáltatások: RPC, RMI

Köztesréteg protokoll (2)



A hálózatos kommunikáció módosított hivatkozási modellje.

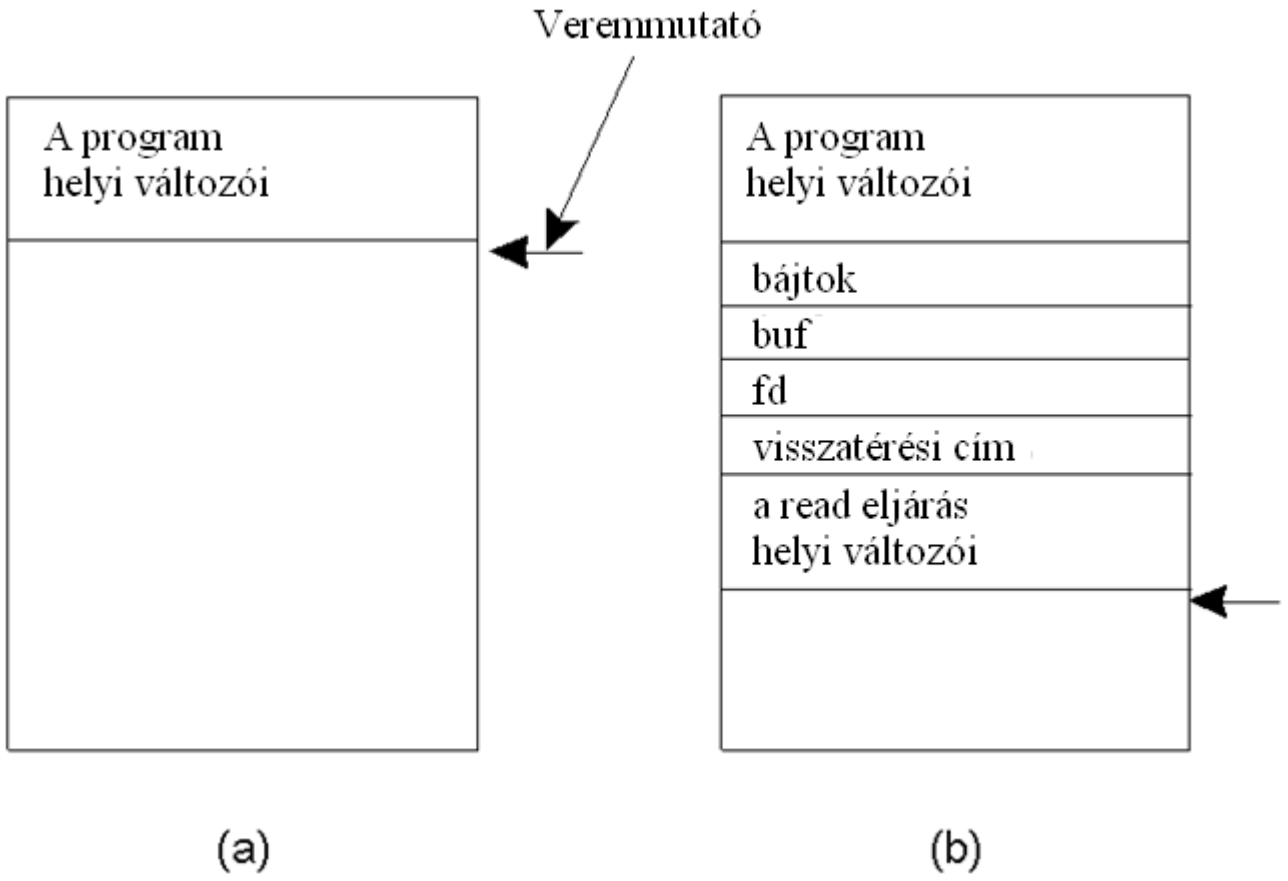
RPC

- Remote Procedure Call
- Elrejti a kommunikáció tényét

- Hagyományos eljáráshívás
  count = read(fd, buf, nbytes);

  - Érték szerinti / hivatkozás szerinti paraméterátadás

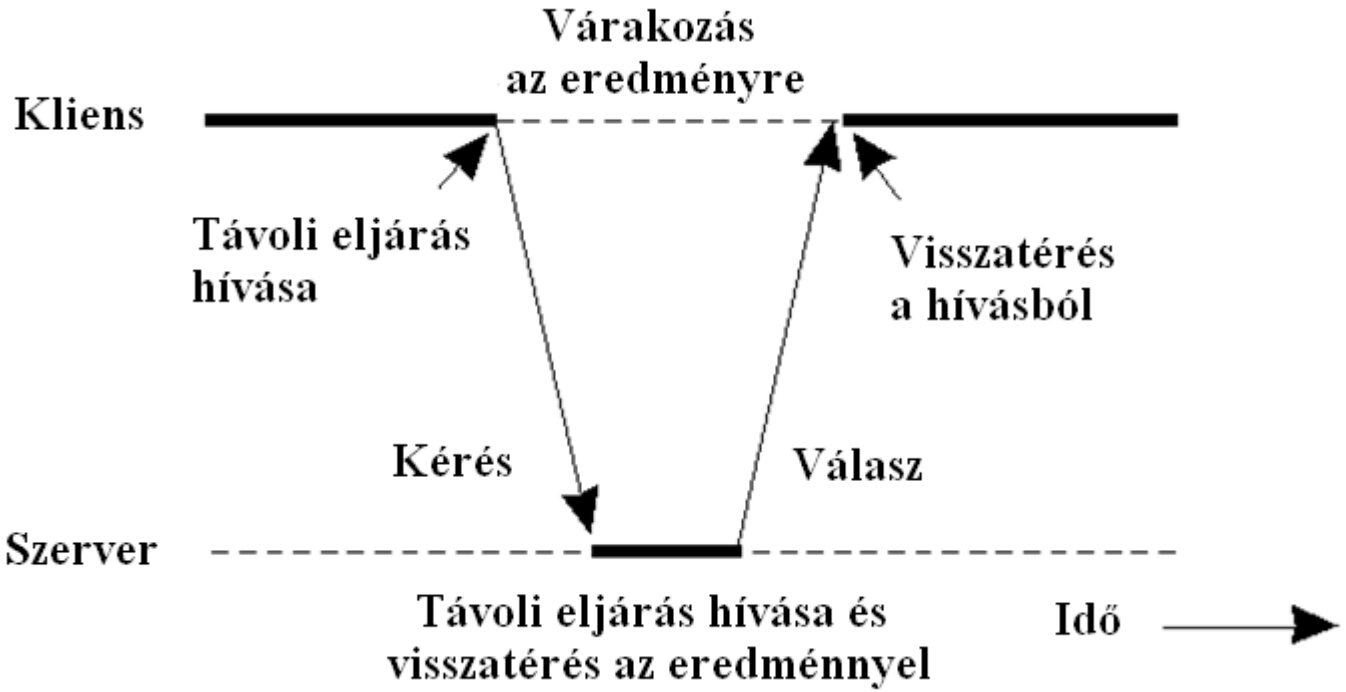
Hagyományos eljáráshívás



a) Paraméter-átadás helyi eljáráshívás esetén: a verem állapota a „read” függvény meghívása előtt

b) A verem állapota a hívott eljárás futása közben

Kliens- és szerveroldali eljáráscsonkok

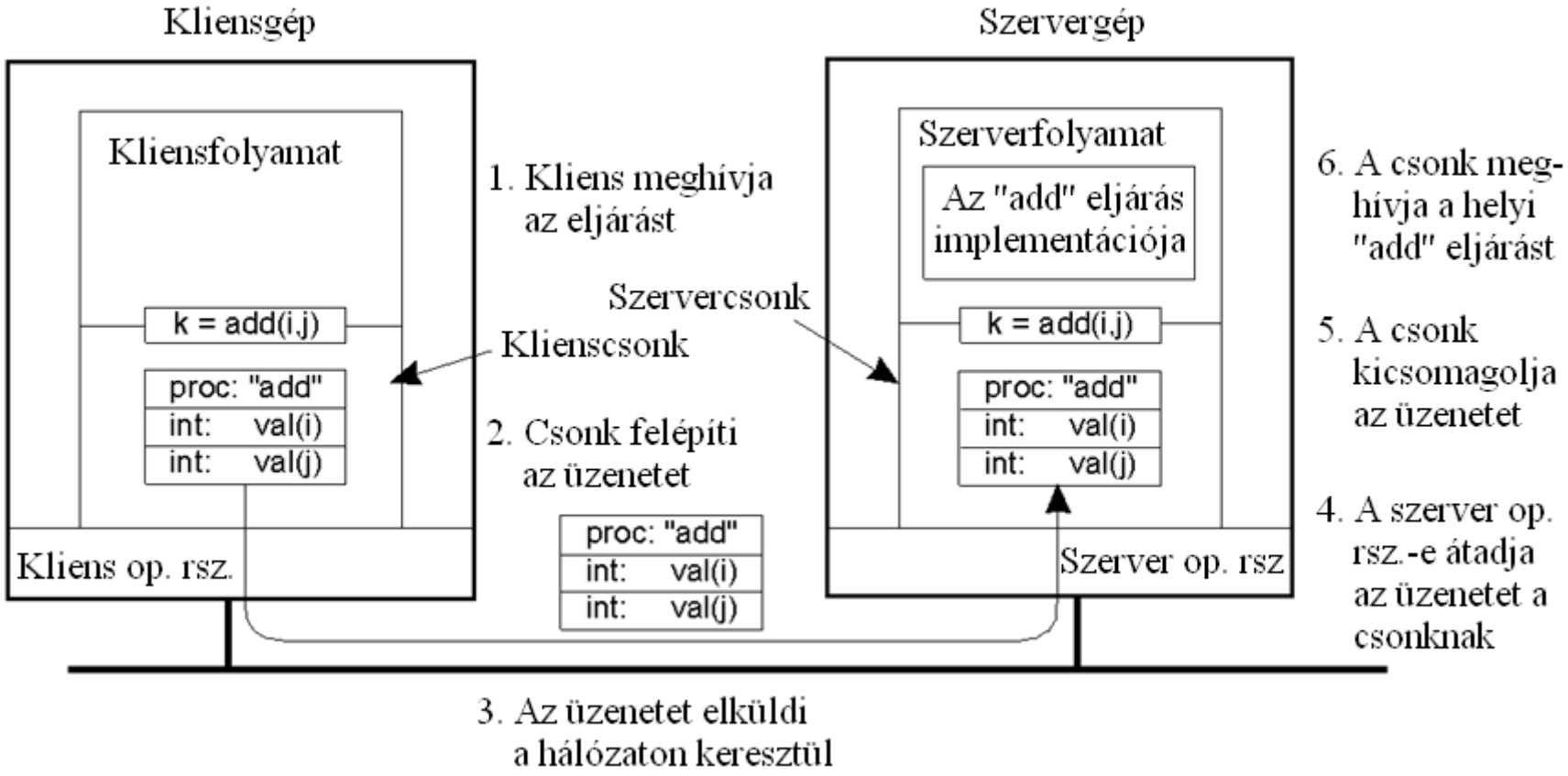


A kliens és a szerver közötti távoli eljárás hívásának (RPC) alapelve.

Távoli eljárás hívása során végrehajtandó lépések

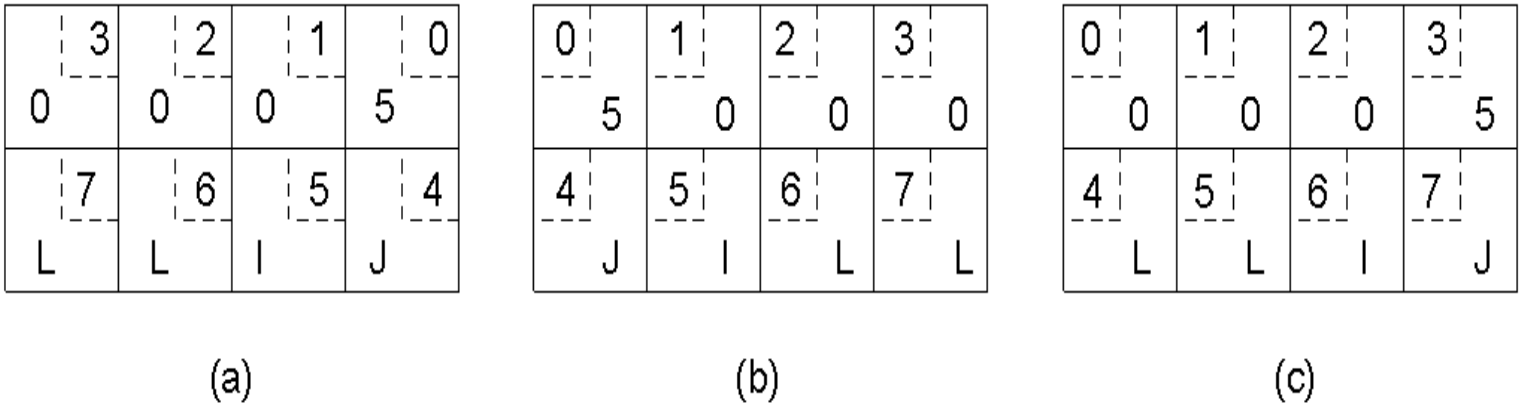
1. A kliensfolyamat hagyományos módon meghívja a kliensoldali eljáráscsonkot.
2. A csonk felépíti az üzenetet, és meghívja a helyi op. rsz.-t.
3. A kliensoldali op. rsz. elküldi az üzenetet a távoli op. rsz.-nek.
4. A távoli op. rsz. átadja az üzenetet a szerveroldali eljáráscsonknak.
5. A csonk kicsomagolja a paramétereket és meghívja a szervert.
6. A szerver elvégzi a feladatát, és visszaadja az eredményt a csonknak.
7. A csonk becsomagolja az eredményt, és meghívja a helyi (szerveroldali) op. rsz.-t.
8. A szerveroldali op. rsz. elküldi az üzenetet a kliensoldali op. rsz.-nek.
9. A kliensoldali op. rsz. átadja az üzenetet a kliensoldali csonknak.
10. A csonk kicsomagolja az eredményt, és visszatér a hívó kliensfolyamathoz

Paraméterek átadása (1)



Az RPC útján történő távoli feladatvégzés során szükséges lépések.

Paraméterek átadása (2)



(a) (b) (c)

a) Az eredeti üzenet a Pentium gépen.

b) Az üzenet, miután megérkezik a SPARC gépre.

c) Az invertált üzenet (a dobozok felső sarkában látható kis számok a bájt sorszámát jelentik).

A paraméterek leírása és a csonk generálása

a) Egy eljárás

b) Az eljárásnak megfelelő üzenet

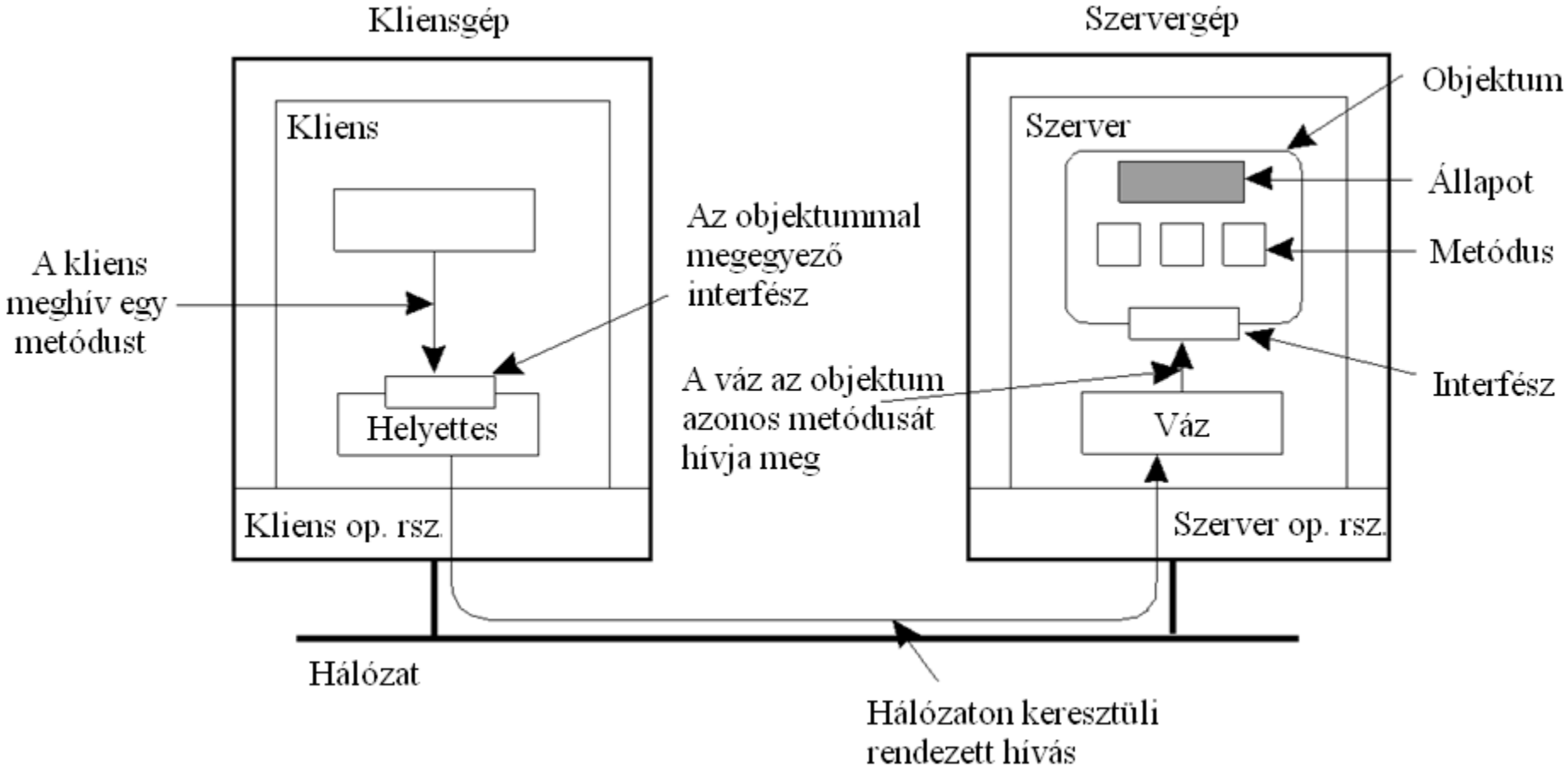
```
foobar( char x; float y; int z[5] )
{
   ....
}
```

(a)

A foobar helyi paraméterei	
	x
y	
5	
z[0]	
z[1]	
z[2]	
z[3]	
z[4]	

(b)

Elosztott objektumok



A távoli objektum elterjedt kialakítása a kliensoldali helyettes (proxy) alkalmazásával