2. fejezet – Hálózati szoftver

Hálózati szoftver és hardver viszonya

Az első gépek összekötésekor (azaz a „hálózat” első megjelenésekor) a legfontosabb lépésnek az számított, hogy elkészüljön az a hardver, ami az összeköttetést fizikailag lehetővé teszi. Amikor a fizikai összeköttetés létrejött, ezután kezdődött a kommunikációt lebonyolító szoftver „kifejlesztése”. A hálózati szoftver tulajdonképpen az egyes gépek operációs rendszerei közötti kapcsolatot biztosította a hálózati hardver megfelelő használatával.
Ma azonban szerencsére már (az idők során kialakult, és jellemzően mindenki által elfogadott) szabványok mentén készül a hardver és a szoftver is. A legtöbb ma is használatban lévő hálózati modell réteges (Layer) szerkezetű. Minden réteg jól meghatározott funkciókért, feladatokért, szolgáltatásokért felelős, és kizárólag a közvetlen alatta illetve felette lévő réteggel tart közvetlen kapcsolatot, csak ezekkel kommunikál. Minden egyes réteg felfogható egy olyan virtuális gépként, ami a felette lévő réteget szolgálja ki. Az egymással szomszédos rétegek között interfész (Interface) található. A különböző hosztok azonos rétegeit társidentitásoknak (Peer) nevezzük. Az azonos rétegek közötti kommunikáció szabályai az úgynevezett protokollok.

Jó és szemléletes példa a réteges szerkezetre és a protokollokra az, ahogyan egy nagy cég ügyvezetőjének a levele eljut a másik nagy cég ügyvezetőjéhez.
Rétegek: Ügyvezető → Titkárnő → Belső postázó → Városi posta → Országos Posta
(és ugyanígy visszafelé is)
Protokollok: (esetünkben szóban és írásban) címzés, diktálás, borítékolás, levél továbbítás
Maga az üzenet (adat) lehet [azért hogy valami hasonlattal magyarázni lehessen] egy levél, egy folytatásos novella, két és fél kiló cukor, videó anyag, stb.

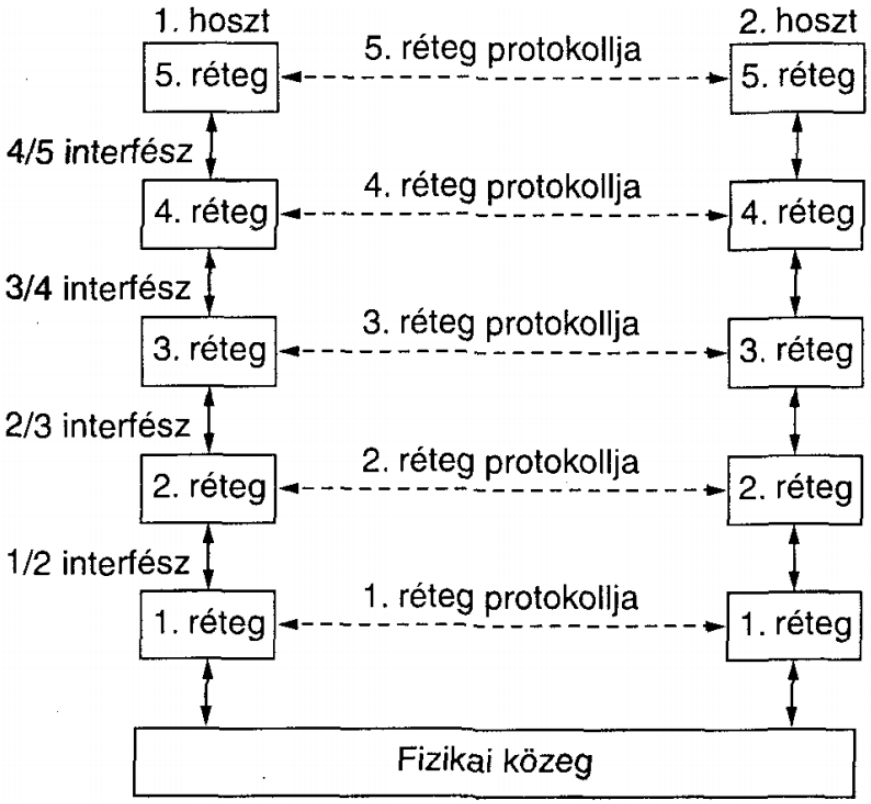
Az egyes rétegek az információt más-más formában dolgozhatják fel illetve továbbíthatják.

Az azonos szintű rétegek szükségképpen rendelkeznek „küldés” vagy „adás” illetve „vétel” nevű eljárással, miközben nem is közvetlenül egymással, hanem az alattuk lévő rétegek segítségével érik el egymást.

Fogalommagyarázat összefoglalva

Interface: Egy hoszton belül az egymás alatti rétegek kommunikációja, ami „valódi” kommunikáció

Protokoll: Különböző hosztok azonos szintjei közötti kommunikáció, ami „virtuális” kommunikáció, hiszen nem közvetlenül, hanem közvetítő rétegek segítségével valósul meg.



A rétegek működésének megbízhatósági és erőforrás-kiosztási kérdéseiről nem szabad megfeledkezni, de ezeket később tárgyaljuk részletesen. (hibajelzés, hibajavítás, útválasztás, címzés; forgalomszabályzás, valós idejű továbbítás, QoS; titkosítás, hitelesítés)

Összeköttetés alapú és összeköttetés nélküli szolgáltatások

Összeköttetés alapú (Connection Oriented Service) rendszer modellezhető például a klasszikus (analóg) telefonhálózattal, ami ebből a szempontból egy csőként viselkedik. Ami az egyik végén bemegy, az (szinte azonnal) ki is jön a másik végén. Nyilván egy ilyen összeköttetés csak akkor működhet, ha a küldő és a fogadó alhálózat megegyezik (Negotiation) a kommunikáció paramétereiben: üzenet maximális hossza, minőség, stb.

Az összeköttetés nélküli (Connectionless Service) rendszerek modellezésére a klasszikus postai levélszolgáltatás a szemléletes példa. Nyilván itt is szempont a levél fizikai mérete (most tekintsük el az üzenet feldarabolásától). Minden levél tartalmazza a címzett adatait, így ha kerülő úton is (mert pl. rossz helyre kerül) előbb-utóbb megérkezik. Az is előfordulhat, ugyanahhoz a címzetthez egy később feladott levél előbb érkezik meg, mint egy korábban feladott levél. Fontos szempont a biztonság, azaz, hogy az üzenet tartalmi változások nélkül, és a megfelelő helyre érkezzen meg. Ezt a fogadó fél nyugtázhatja, amiről a feladó értesül. Ez természetesen plusz idő, plusz kommunikációt jelent, de van, amikor a visszajelzés elengedhetetlen.

A megbízható összeköttetés alapú szolgáltatás egyik tipikus alkalmazása a fájl (file) átvitel. Nyilván csak a hibamentes működés fogadható el, egy bitet sem veszíthetünk.
A megbízható összeköttetés alapú szolgáltatásnak két altípusa van: az üzenetsorozat (ami megtartja az üzenethatárokat, tehát az üzenetei nem olvadnak egybe) és a bájtfolyam (ami ömlesztve viszi át az adatokat).

	Szolgáltatás	*Példa*
Összeköttetés alapú	Megbízható üzenetfolyam	Könyvlapok (fájlok) sorozata
	Megbízható bájtfolyam	Filmletöltés
	Megbízhatatlan összeköttetés	Internetes telefonálás (VoIP)
Összeköttetés nélküli	Megbízhatatlan datagram	Kéretlen levél generálása
	Nyugtázott datagram	Szöveges üzenetküldés
	Kérés – Válasz	Adatbázis lekérdezés

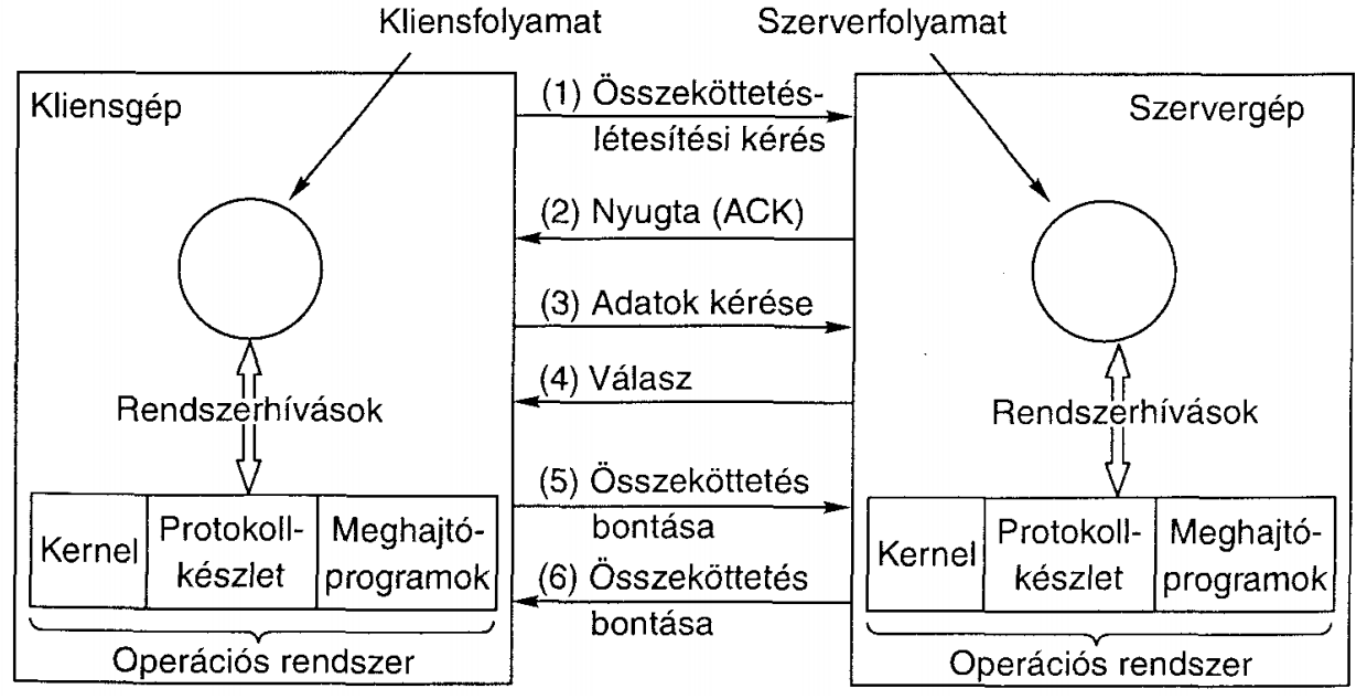
Szolgáltatási primitívek

A primitívek – azaz elemi műveletek – az „operációs rendszerek c. tantárgy” során megismert rendszerhívások az egyes szolgáltatások eléréséhez. Amennyiben a protokollkészlet az operációs rendszerben található (jellemzően ott található), akkor a primitívek a már megismert [megtanult vagy kipuskázott (?)] módon kernel módba kényszeríti a számítógépet, és így el tudja kezdeni az adatok küldést vagy fogadását.

Természetesen a szolgáltatás különböző módjai más-más primitív készletet igényelnek.

Hat szolgáltatási primitív egy egyszerű összeköttetés alapú szolgáltatás megvalósításához	
Primitív	*Jelentése*
LISTEN	Blokkolt várakozás bejövő kapcsolatfelvételre
CONNECT	Összeköttetés létrehozása egy várakozó társidentitással
ACCEPT	Bejövő kapcsolat fogadása egy társidentitástól
RECEIVE	Blokkolt várakozás bejövő üzenetre
SEND	Üzenet küldése a társidentitásnak
DISCONNECT	Összeköttetés bontása

Életszerű példa ennek a kommunikációs módnak a szemléltetésére például egy éjjel-nappal üzemelő, telefonos hiba bejelentéseket fogadó szolgáltatás, hiszen a kezelő (ha jól végzi a dolgát, akkor) folyamatosan azt figyeli, hogy érkezik-e bejövő hívás. Amint befut egy hívás, a kezelő a hívás fogadásával gyakorlatilag részéről mindent megtett a kapcsolat felépítésért. Amint a kapcsolat felépült elkezdődik a beszélgetés, azaz az adat/információ csere. A beszélgetés végeztével a kapcsolat bontása is megtörténik.
Egy általános, összeköttetés alapú, egyszerű Kliens-Szerver kommunikáció, amely nyugtázott datagram szolgáltatás segítségével valósul meg, a következő ábra alapján érthető meg :



A legelső lépés, azaz a kezdeti állapot az, amikor a szerver (azaz az adó) oldalon egy LISTEN primitív folyamatosan figyel, hogy érkezik-e valamilyen kapcsolat felvételi igény. A kapcsolat felvételi igény beérkezését a kliens (az a vevő) oldal a CONNECT primitívjének a megérkezése jelzi. A CONNECT primitívnek egy címzést mindenképpen tartalmaznia kell, hogy csak a kívánt (azaz a megcímzett) szerver válaszoljon a kérésére. A megfelelő címzés ismerete teszi lehetővé, hogy a fizikai kapcsolaton túl, egy logikai kapcsolat is felépülhet. Az adó oldalon a CONNECT primitív kapcsolatépítésének hatására – ideális esetben – elindul az ACCEPT (Acknowledgement) primitív, amely az adó oldal részéről jelzi a kapcsolatépítési szándék elfogadását. Ezek után kezdődik a felhasználó szempontjából hasznos adatátvitel a SEND illetve a RECEIVE primitívek segítségével. A kommunikáció lezárásaként jellemzően a vevő oldal – miután minden szükséges adatot megkapott – kezdeményezi a kapcsolat bontását egy DISCONNECT primitív elindításával, amire az adó oldal szintén egy DISCONNECT primitívvel válaszolva bontja a kapcsolatot, azaz lezárja a kommunikációt.
Érdekes észrevétel, hogy az összeköttetés alapú protokoll esetében az összeköttetés megvalósításához, lebonyolításához minimum 6db átvitt csomagra (Connect, Accept, Send, Receive, Disconnect, Disconnect; mivel a Listen nem része az átvitelnek) van szükség, amennyiben a felhasználó szempontjából hasznos információ egyetlen SEND/RECEIVE segítségével lebonyolódik. Nagyobb mennyiségű adat továbbításakor felléphetnek átviteli hibák, elveszhetnek csomagok, melyek ismétléséről gondoskodnunk kell, illetve tudnunk kell azonosítani az utolsó (záró) csomagot is.

Egy összeköttetés nélküli átvitel esetén, optimális esetben, elvben akár csupán két csomag folyamatos és ciklikus igénybevételének a segítségével (SEND, RECEIVE) megoldható az adatok átvitele. Ez esetben nyilván minden egyes csomagnak tartalmaznia kell a megfelelő címzést. A csomagok ezen kívül kell hogy a saját azonosítójukat (sorszámukat) is tartalmazzák, ami a helyes sorrend, illetve az utolsó, azaz a záró csomag azonosításához elengedhetetlen.
Így első látásra az összeköttetés nélküli protokoll sokkal egyszerűbbnek tűnik mint az összeköttetés alapú protokoll, de ez csak a látszat, hiszen számos olyan gyakorlati probléma kezelését is valahogyan meg kell oldani, ami egy összeköttetés alapú protokoll esetében fel sem merül (pl. a csomagok sorrendje).

Mindkét szolgáltatás esetben (összeköttetés alapú, illetve összeköttetés nélküli) a felhasználó szempontjából hasznos adatok átvitele mellett további adatcserére is szükség van, hiszen a rideg valóság közel sem modellezhető ideális laborkörülmények között. A csomagok elveszhetnek, késhetnek, hibásan vagy rossz sorrendben érkezhetnek, stb.

A szolgáltatások kapcsolata a protokollokkal

A szolgáltatás és a protokoll könnyen összekeverhető fogalmak.
(Újabb) fogalommagyarázat
Szolgáltatás: Azon primitívek (tehát elemi műveletek) halmaza, amelyeket egy adott réteg a felette lévő rétegek számára biztosít. A szolgáltatás azt definiálja, hogy egy adott réteg a felhasználó nevében milyen műveleteket képes végrehajtani, de azt nem hogy hogyan. A szolgáltatás tehát a két szomszédos réteg közötti interfésszel kapcsolatos, ahol az alsó réteg a szolgáltató, a felső réteg pedig a szolgáltatás felhasználója.
Protokoll: Olyan szabályok halmaza, amely a „hogyan” kérdéssel kapcsolatosak. Azaz hogy milyenek legyenek azok a csomagok, keretek, üzenetek, amelyekkel a társidentitások (Peer) kommunikálnak. Ne felejtsük el, hogy a különböző peer-eken különböző operációs rendszerek esetében is meg kell tudnunk valósítani a kommunikációt.
A protokoll tehát a szolgáltatás implementációjának felel meg, és mint ilyen, láthatatlan a szolgáltatást igénybe vevők számára.

[implementáció: Egy algoritmus, architektúra, szabvány, modell, specifikáció vagy egyéb terv konkrét megvalósítása. Az implementáció a számítástechnikában egy technikai specifikáció vagy algoritmus program, egy program komponens vagy valamely más módon történő megvalósulás.]