Az adatkapcsolati réteg

Számítógép-hálózatok

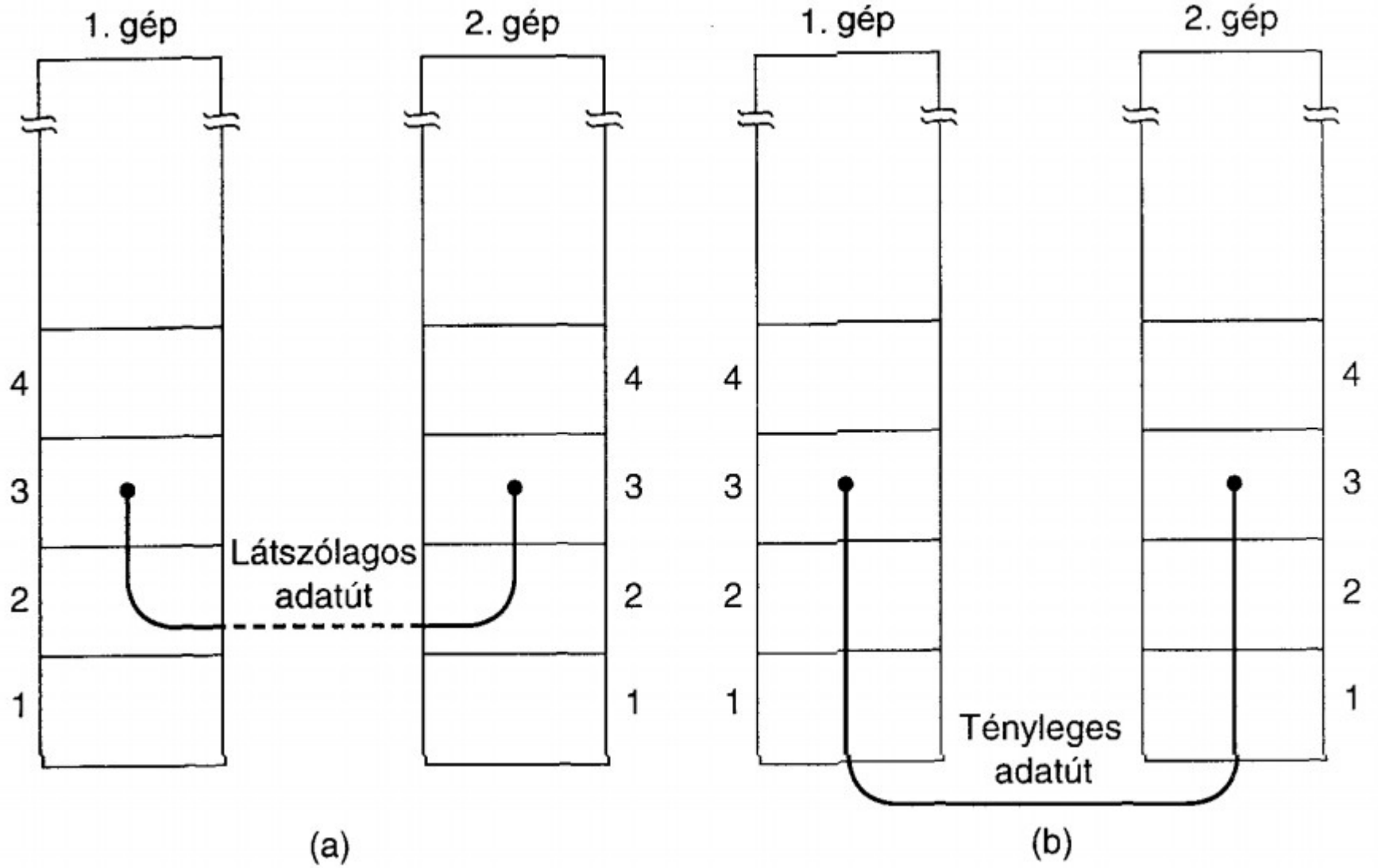
Kajdocsi László
A-602,
kajdocsi.laszlo@sze.hu

A réteg fő feladatai

Biztosítja az adatok biztonságos eljutását az adótól a vevőig:

- **Azonosítható adatkeretek**
- **Vezérlőbitek**
- **Nyugtázás**
- **Forgalomszabályozás**

Rétegek közötti kommunikáció



Hogy működik valójában?

Hálózati<->Adatkapcsolati<->Fizikai réteg

- **Az adó hálózati rétegéből érkező bitfolyamokat diszkrét keretekké alakítja az AK réteg + ellenőrző jelekkel látja el**
- **A bitfolyam kereteket fizikai jelfolyammá alakítja a F réteg majd juttatja a vevő F rétegébe**
- **A vevő F rétege vissza alakítja a jelfolyamokat bitfolyamokká, majd az AK réteg rendezi a kereteket és továbbítja a H rétegnek**

A hálózati rétegnek nyújtott szolgáltatások

- **nyugtázatlan összekötés nélküli szolgálat** (Nincs előzetes kapcsolat építés vagy bontás: pl. Ethernet vagy Spam levelek)
- **nyugtázott összekötés nélküli szolgálat** (Nincs előzetes kapcsolat építés, de nyugtázott keretek vannak: pl. WLAN, vagy szöveges üzenet)
- **nyugtázott összekötés alapú szolgálat** (Előzetes kapcsolatépítés, megbízható átvitel: pl. fájlátvitel)

Keretezés feladatai

- **Karakterszámlálás, bájtszámlálás**
- **Kezdő- és végkarakterek beszúrása**
- **Kezdő- és végbitek beszúrása**
- **Fizikai réteg kódolásának megsértése**

Hibakezelés (Error Control)

- **Pontosan egy megérkezés (időzítők, számlálók kezelése)**
- **Pozitív vagy negatív nyugta a küldőnek (visszacsatolás)**
- **Ismétléssel történő javítás (újraküldés)**

Forgalomszabályozás (Flow Control)

Gyorsabb a küldés, mint a fogadás. Mit tehetünk?

1) Visszacsatolás-alapú forgalomszabályozás

- ✔A vevő engedélyt küld a további adatok küldésére

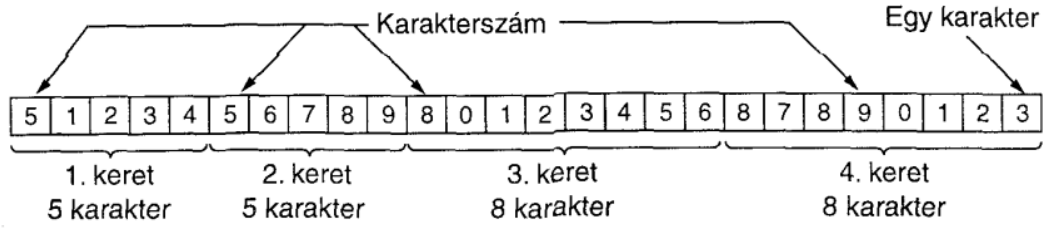
2) Sebesség-alapú forgalomszabályozás

- ✔Beépített sebességkorlát, amit a küldő tiszteletben tart
- ✔Nincs visszacsatolás a fogadótól
- ✔Az AK réteg ezt nem használja

Keretezés

- **Küldő oldalon keretekbe tördelés**
- **Fogadó oldalon a keretek visszafejtése**
- **Az entitások alapegysége az Adat Keret (Data Frame)**
- **Fontos! A keret ellenőrző összeget tartalmaz, ami jelzi, hogy épségben megérkezett-e a keret**
- **Ha eltérés van az ellenőrző keretben, akkor meg kell ismételni a küldést**

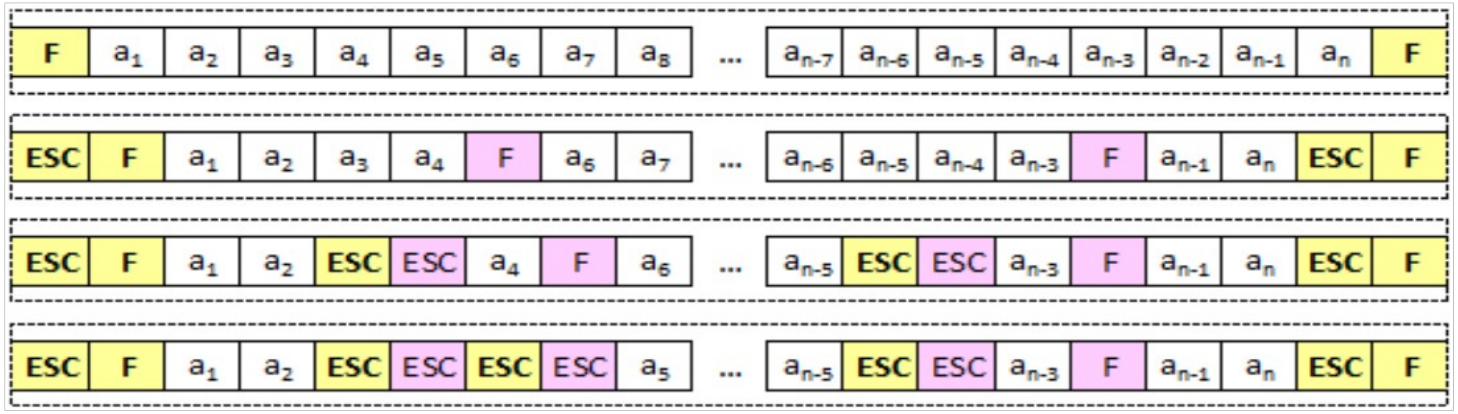
Karakterszámlálás, bájtszámlálás



- **1 karakter = 1 bájt**
- **A keret fejlécébe, azaz bájtszámmezőbe írja a karakterek számát**
- **A vevő AK rétege a bájtszámmezőből kiolvassa a keret hosszát**
- **Hibalehetőség: átviteli hiba esetén pont a bájtszámmező kap rossz értéket, így átláthatatlanná válik a keretfolyam**
- **Ma már jellemzően nem használják!**

Kezdő és végkarakterek beszúrása

- **Keret elejét és végét egy jelzőbájttal, ún. flaggel jelöljük meg**
- **Gyakorlatban egyforma flageket használnak**
- **Szinkronizációs hiba esetén csak meg kell keresni a jelzőbájtot és megtaláljuk a küldés alatt levő keret végét**
- **Byte stuffing**



Kezdő és végbitek beszúrása

- **A bájtbeszúrás felesleges adattal terheli a csatornát. Mi a megoldás?**
- **<u>Bitbeszúrás:</u> tetszőleges számú bit legyen a keretben, a karakterkódok is tetszőleges számú bitből állnak**
- **Speciális bitmintával indul a keret: 6 db 1-es (EZ A FLAG)**
- **Az adó oldalon az AK réteg öt db egymást követő 1-es után egy 0-t szúr be, vevő oldalon a fordítottja történik, minden öt db 1-es után töröl egy 0-t**

Fizikai rétegbeli kódolás megsértése

- **A fizikai réteg redundanciáját használjuk ki ennél a kódolásnál**
- **4B/5B kódolás: 4 bites adatcsoportokat kódol 5 bites adatcsoportokba (2^5 jelsorozatból csak 2^4 van használatban, így a többit kódolásra használhatjuk)**
- **Olyan kód is átvitelre kerül, ami esetben nem kerülhetne, ezért kódsértés**
- **Itt van a legkevesebb felesleges adat átvitel közben**

A gyakorlatban

- **A legtöbb AK protokoll a nagyobb biztonság érdekében a fent említett módszerek kombinációját alkalmazza**
- **Legtöbb protokollban a keret elején egy előtag van (ún. Preamble), ezt követi egy hosszújábot jelző bájt**
- **A keret tulajdonságai biztosítottak ezáltal**

THE END

Köszönöm a figyelmet!